1996-06

Geometric formation with uniform distribution and movement in formation of distributed mobile robots

Alptekin, Gokhan

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

PLRS AND EPLRS: A CASE STUDY IN SYSTEM DEVELOPMENT AND POST DEPLOYMENT SOFTWARE SUPPORT

by

Jon K. Aldridge

December 1996

Principal Advisor: Martin J. McCaffrey

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE December 1996	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE **PLRS AND EPLRS: A CASE STUDY IN SYSTEM DEVELOPMENT AND POST DEPLOYMENT SOFTWARE SUPPORT**

5. FUNDING NUMBERS

6. AUTHOR(S) Aldridge, Jon K.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT *(maximum 200 words)*

Software development and acquisition have been the Achilles' heel within the Department of Defense for many years. In spite of considerable oversight and the control exercised by many regulations and standards, there still exists significant problems in cost, schedule, and delivered capability within programs. This thesis looks at the acquisition of two software and firmware intensive programs, the Position Location and Reporting System (PLRS) and the Enhanced PLRS (EPLRS). Its primary focus is the transition of life cycle management of the software to the government post deployment software support (PDSS) activity. The acquisition of PLRS by the U.S. Marine Corps involved the acquisition of an unprecedented new technology and system capability never before attempted. As a result, the configuration management, testing, and transfer of the software maintenance support functions caused considerable problems at the PDSS activity. A number of the lessons from this experience were applied to the acquisition and development of the Army's EPLRS resulting in a more thorough statement of contractual requirements for the contractor, better understanding of the configuration management by the government, and the testing of the system under more realistic conditions to validate its abilities. The recommendation of this thesis will result in a smoother acquisition process, a more mature system at time of delivery to the government, and a more capable PDSS.

14. SUBJECT TERMS PLRS, EPLRS, System Development, Post Deployment Software Support			15. NUMBER OF PAGES 147
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

Approved for public release; distribution is unlimited.

PLRS AND EPLRS: A CASE STUDY IN SYSTEM DEVELOPMENT AND POST DEPLOYMENT SOFTWARE SUPPORT

Jon K. Aldridge
Major, United States Marine Corps
B.B., Western Illinois University, 1982

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
December 1996

Author: [redacted]
Jon K. Aldridge

Approved by: [redacted]
Martin J. McCaffrey, Principal Advisor

[redacted]
Magdi Kamel, Associate Advisor

[redacted]
Reuben T. Harris, Chairman
Department of Systems Management

ABSTRACT

Software development and acquisition have been the Achilles' heel within the Department of Defense for many years. In spite of considerable oversight and the control exercised by many regulations and standards, there still exists significant problems in cost, schedule, and delivered capability within programs. This thesis looks at the acquisition of two software and firmware intensive programs, the Position Location and Reporting System (PLRS) and the Enhanced PLRS (EPLRS). Its primary focus is the transition of life cycle management of the software to the government post deployment software support (PDSS) activity. The acquisition of PLRS by the U.S. Marine Corps involved the acquisition of an unprecedented new technology and system capability never before attempted. As a result, the configuration management, testing, and transfer of the software maintenance support functions caused considerable problems at the PDSS activity. A number of the lessons from this experience were applied to the acquisition and development of the Army's EPLRS resulting in a more thorough statement of contractual requirements for the contractor, better understanding of the configuration management by the government, and the testing of the system under more realistic conditions to validate its abilities. The recommendation of this thesis will result in a smoother acquisition process, a more mature system at time of delivery to the government, and a more capable PDSS.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

Throughout the 1980s and 1990s many Department of Defense (DoD) tactical weapons, command, control and communications and intelligence systems have been procured that significantly depend upon software to function. Software has become critical to our war fighter's ability to fight. Unfortunately, a large number of these weapon systems have been cited as having significant software development problems that include cost and schedule overruns and not meeting user requirements. In the past several years DoD has been cited in numerous General Accounting Office (GAO) reports (C-17 transport aircraft, Army Fire Direction Data Manager, and AN/FQ-93 NORAD radar) with inadequate management attention, ill-defined system requirements, and inadequate testing of its software and weapon systems. In response, DoD has attempted to solve these problems by reviewing and restructuring the acquisition and oversight process. The Defense Management Report of 1989 and the draft DoD Software Master Plan of 1990 addressed a number of these problems. In future years, the DoD will be hard pressed to justify the continued acquisition of weapons with mission critical software without assurances that the process is well managed, systems thoroughly tested and requirements well defined.

A. AREA OF RESEARCH AND OBJECTIVE

This thesis will focus on the development, transition and post-deployment software support (PDSS) issues associated with the Position Location Reporting System (PLRS) and the lessons learned for the Enhanced Position Location Reporting System (EPLRS). The Marine Corps fielded PLRS in 1987; EPLRS is in development for the U.S. Army. This research concentrates on the lessons learned in fielding the PLRS software and how these lessons are being applied to the EPLRS software acquisition.

B. RESEARCH QUESTIONS

1. What were the problems and shortfalls associated with the transition of PLRS software from the contractor to the government software support activity?

2. What action was taken for EPLRS to rectify problems identified with the transition of PLRS software for PDSS?

3. What action was taken to rectify the problems more recently identified during developmental testing with Enhanced PLRS software?

C. SCOPE

The scope of this study is limited to the software related issues associated with the transition of PLRS from the contractor to the government, and the issues, to date, associated with EPLRS software. As previously mentioned, PLRS was first fielded to the government in 1987. EPLRS is currently in the Low Rate Initial Production (LRIP) phase for the Army. Hughes Aircraft Company (HAC) is the developer and manufacturer of both systems. Where functionality is the same the systems are compatible and have many similarities. Lessons learned during the fielding, development, and testing phases of PLRS software have been applied in the EPLRS program. The Marine Corps Tactical Systems Support Activity (MCTSSA) Camp Pendleton, California is the post-deployment software support activity (PDSSA) for both PLRS and EPLRS. This thesis will document many of the lessons learned by MCTSSA from these two procurements.

D. METHODOLOGY

The research methodology consisted of a literature review and interviews (both face-to-face and by telephone) with government and industry officials associated with PLRS and EPLRS.

1. Literature Search

A literature search was conducted to get background material concerning laws, DoD regulations, and policies dealing with the acquisition of Mission Critical Computer Resources (MCCR). GAO reports, government and non-government publications were reviewed. One by-product of this review was a compilation of difficult to find sources from a variety of organizations, libraries, and repositories. This time-consuming research should serve as a valuable beginning resource for any future studies in this subject area. Specific literature dealing with PLRS and EPLRS came from MCTSSA.

2. Interviews

Personal and telephone interviews were conducted with individuals from the government program manager's office at Fort Monmouth, New Jersey, the PDSSA, and individuals involved in the software development effort at HAC. It included both individuals currently associated with the program, and formerly involved with the systems at various times in the acquisition process. Appendix A contains a list of interviewees. These participants spoke frankly on their personal perspectives of the issues. Without their open discussions and generous cooperation this research could not have been completed.

E. ORGANIZATION

This thesis is divided into six chapters. Chapter II provides an overview of the relevant literature dealing with the acquisition of MCCR software. It will also give a brief review of acquisition regulations that guide how the government acquires software systems. Service specific regulations on acquiring embedded software for the DoD are also discussed.

Chapter III provides a history of the PLRS and EPLRS programs. It discusses the events, as well as key milestones, in the acquisition of these systems. To provide

a background on the systems, the mission of PLRS and EPLRS and the tactical employment of the systems are discussed. This chapter also gives a general overview of the software, hardware and firmware associated with PLRS and EPLRS.

Chapter IV addresses the software and firmware specific issues that caused problems and were solved during the acquisition and PDSS of PLRS. The lessons learned from the acquisition and support of PLRS that were useful to EPLRS are also discussed.

Chapter V reviews the EPLRS program, much as Chapter IV looked at PLRS. Although EPLRS has not been fielded, a number of the PLRS lessons learned have been applied to the EPLRS program.

The research questions posed in this chapter are answered in Chapter VI. Conclusions from this research concerning the future of software acquisition for embedded computer systems are made. Recommendations for areas that may warrant future research efforts are also included.

II. MILITARY SOFTWARE DEVELOPMENT HISTORY

Billions of dollars are spent every year within the DoD for MCCR. Over the years a large number of policies, standards and guidance regarding software and systems has developed. The initial focus of this chapter is on those regulations that effect the acquisition process in general and also impact software development. The chapter covers five areas: legislative actions, DoD procurement guidance, Department of the Navy (DoN) procurement guidance, Tactical Digital Standards, and government specifications and standards. Additional sections cover the acquisition system process with an emphasis on the software life cycle. A specific review of a software intensive system procurement is made.

A. LEGISLATIVE ACTIONS

These are two major legislative acts that must be considered when procuring MCCR. A short discussion of each follows.

1. Public Law 89-306 (Brooks Act) of 1965

This law, commonly referred to as the "Brooks Act," promotes competition and insures stability in the procurement of Automated Data Processing (ADP) resources. This bill forces federal agencies to analyze their ADP requirements and then through competition procure the most economic and efficient system. It assigns responsibility in procuring ADP resources (both hardware and software) for federal agencies to the General Services Administration (GSA), policy guidance and overall leadership to the Office of Management and Budget (OMB), and ADP standards to the National Institute of Standards (formerly the National Bureau of Standards). Realizing it could not be the sole procuror of all ADP resources for all the federal government, GSA has provided a limited delegation of procurement authority that varies from agency to agency. This law does not permit the GSA to determine an organization's ADP

requirements. The agency determines its requirements for ADP equipment and the method of procurement and if it exceeds an agency's blanket delegation, GSA must approve the method of procurement. The DoD considers MCCR exempt from the Brooks Act because provisions exclude embedded systems. [Ref. 1]

2. Public Law 97-86 (Warner Amendment) of 1982

This amendment was implemented in order to broaden and provide more detailed guidance on the range of embedded computer resources that could be excluded from the provisions of the original Brooks Act. It distinguishes the DoD's mission critical systems from business oriented automated information systems. It defines mission critical systems as those that fall into the following four categories: related to intelligence and cryptologic missions; provide command and control of military forces; are integral to a weapon system (embedded systems); or are critical to fulfilling military or intelligence missions. Systems used for routine administrative and business applications, such as a logistical system are not exempt. Doubt as to the applicability of this law is determined by the Under Secretary of Defense (for Acquisition). [Ref. 1]

B. DEPARTMENT OF DEFENSE PROCUREMENT GUIDANCE

This section covers DoD and specific Navy and Marine Corps guidance. Both PLRS and EPLRS have contract awards that were made prior to major documentation revisions and consolidations. Therefore, requirements in some earlier publications are the ones that contractually apply to these two systems.

1. DOD Directive 5000.1

Department of Defense Directive 5000.1, dated 23 February 1991 and titled "Defense Acquisition," is first in precedence among DoD directives for providing policies and procedures for managing acquisition programs (except in cases when statutory provisions override). This policy governs defense acquisition by DoD

components. The directive is divided into three areas: policies governing defense acquisition, an integrated management framework, and responsibilities of key individuals.

Part One of the directive provides guidance in translating operational needs into stable and affordable programs, acquiring quality products, and organizing for efficiency and effectiveness. Part Two of the directive describes the integration of the requirements generation system, acquisition management system, and the planning, programming, and budgeting system (PPBS). It describes the characteristics of each system and highlights the relationships that must be maintained for effective decision making. Part Three describes the acquisition responsibilities of key officials and forums. It starts with the Deputy Secretary of Defense and continues through the program manager (PM). [Ref. 2]

2. DOD INSTRUCTION 5000.2

Department of Defense Instruction 5000.2, dated 23 February 1991 and titled "Defense Acquisition Management Policies and Procedures," provides a core of fundamental policies and procedures that can be implemented by the PM. The instruction has consolidated information condensed from 45 DoD directives and instructions that were canceled, as well as various DoD component publications that were also canceled. The instruction provides guidance in a variety of areas that include: requirements evolution and affordability; acquisition planning and risk management; engineering and manufacturing; logistics; test and evaluation; configuration and data management; management and contracts; program control and review; and the Defense Acquisition Board (DAB) process. Section D of part 6 deals with computer resources that were formerly covered in DoD Directive 5000.29. [Ref. 3]

C. DEPARTMENT OF THE NAVY PROCUREMENT GUIDANCE

There are several DoN publications that address MCCR issues. A short review follows.

1. SECNAVINST 5200.32

Secretary of the Navy Instruction 5200.32, dated 11 June 1979 and titled "Management of Embedded Computer Resources in Department of the Navy Systems," implemented DOD Directives 5000.29 and 5000.31 within the DON at the time of the PLRS award. It supplemented policies and procedures for management of Navy weapons and communications, command, control, and intelligence systems when embedded computer resources were incorporated as integral components. Section D of Part 6 (Computer Resources) in DOD Instruction 5000.2 incorporated the requirements from DOD Directive 5000.29. This instruction established a Management Steering Committee for Embedded Computer Resources (MSC-ECR) to oversee and coordinate the accomplishments of policy. This directive was applicable to the PLRS procurement. [Ref. 4]

This June 1979 version of the instruction was superseded in May 1993. The current version includes all of the relevant information from Tactical Digital Standards (TADSTANDs) A and B. For this discussion, the older version, that was applicable during the procurement, is referenced. [Ref. 5]

2. OPNAVINST 5200.28

Office of the Chief of Naval Operations Instruction 5200.28, dated 25 September 1986, and titled "Life Cycle Management of Mission-Critical Computer Resources (MCCR) for Navy Systems Managed Under the Research, Development, and Acquisition (RDA) Process" applied to the PLRS procurement. It covers all MCCR, and includes software that is an integral part of a weapons, command and control, communications, intelligence or other tactical or strategic system aboard

ships, aircraft, and shore facilities, and their support systems. This instruction addresses a Standard Embedded Computer Resources (SECR) program, objectives of which are to: support and improve existing systems, establish a common approach to the acquisition process, plan for the evolution of SECR to meet future needs, and achieve an economy in logistics support, training, manpower, and development. This instruction also mandates a Computer Resources Life Cycle Management Plan (CRLCMP) as a planning document for computer resources throughout the system life cycle. [Ref. 6]

This instruction has recently been superseded by a more current version. For the purpose of this case the 1986 version is referenced because of its applicability to the program. [Ref. 5]

3. MCO 5200.23A

Marine Corps Order 5200.23A, dated 30 December 1986 and titled "Management of Mission Critical Computer Resources in the Marine Corps," implements policies directed by the Secretary of the Navy for the Marine Corps. The scope of this document includes embedded computer resources (ECR) required for tactical data systems (TDS). The purpose of this order is to reduce the life cycle support costs of all Marine Corps mission critical systems by reducing the proliferation of mission critical system hardware and software. This order emphasizes the goal of developing or selecting software and hardware to optimize system performance and ensuring maintainability over the entire life cycle of the system. There are several specific requirements of this order that are relevant to this study. The TDS Acquisition Program Sponsor (APS) is required to do the following: designate a system post-deployment software support activity no later than milestone one in the life cycle of the TDS (MCTSSA was designated for PLRS); ensure the completion of the Computer Resources Life Cycle Management Plan (CRLCMP); discourage the use

of machine and assembly languages, and encourage the use of DOD approved higher order languages (e.g., Ada, CMS-2, Jovial). For hardware the requirements are: that the hardware already be in the DOD inventory; be an approved Navy standard computer; and use of commercial hardware was not authorized unless approved by deviation or waiver. The order also outlines the deviation and waiver request requirements. [Ref. 7]

4. Tactical Digital Standards (TADSTANDs)

Tactical Digital Standards (TADSTANDs) complemented OPNAVINST 5200.28 by establishing procedures and waiver requirements that were unique and essential to the area of MCCR. The policies within TADSTANDs were applicable to all Navy mission critical systems that contained computer resources and were applicable to the PLRS acquisition. There are five TADSTANDs.

a. **TADSTAND A**, dated 2 July 1980, titled "Standard Definitions for Embedded Computer Resources in Tactical Digital Systems," establishes standard definitions for embedded computer resources in tactical digital systems [Ref. 8].

b. **TADSTAND B**, dated 2 January 1985, titled "Computer Hardware, Peripheral, and Interface Standards for Mission-Critical Systems," dictates policy and standards for hardware, peripherals, and interfaces in mission critical systems. This TADSTAND applies to the system from initial concept exploration to post-deployment support. [Ref. 9]

c. **TADSTAND C**, dated August 1990, titled "Computer Programming Language Standard Policy for Mission Critical Computer Resources," cites Ada as the Navy's standard programming language for mission critical systems. The specific requirement for Ada exists unless there are overriding cost, schedule or performance considerations. This TADSTAND also requires that all new develop-

ment projects and major upgrades of existing systems implement Ada. This TADSTAND did not affect the PLRS development. [Ref. 10]

d. **TADSTAND D**, dated 27 October 1989, titled "Reserve Capacity Requirements for Mission Critical Systems," provides policy and procedures for required reserve capacities in MCCR hardware and firmware to accommodate the future growth of operational requirements and growth not known at the time of the acquisition, development, or upgrade of the system. This TADSTAND covers five specific areas: main memory, secondary storage, processor throughput, number of input/output channels, and input/output channel throughput. The specific requirements of this TADSTAND were not applicable to the PLRS acquisition, less stringent requirements in this area were required. [Ref. 11]

e. **TADSTAND E**, dated 24 January 1989, titled "Software Development Documentation and Testing Policy for Navy Mission Critical Systems," details several unique Navy requirements omitted from DoD-STD-1679A. This TADSTAND details additional software testing and acceptance requirements. It was not a requirement for the PLRS acquisition. [Ref. 12]

These TADSTANDs were superseded as of May 1993. The current version of SECNAVINST 5200.32 and SECNAV Notice 5200 contain the requirements that were previously in TADSTANDs A and B respectively. The Ada waiver requirement of TADSTAND C is contained in SECNAVINST 5200.34. The guidance found in TADSTANDs D and E, while not official requirements, remain good policy and guidance until included in the Navy's Program Manager Guidebook or when future instructions are issued. [Ref. 5]

D. SPECIFICATIONS AND STANDARDS

There were several DoD and military standards (MIL standards) that existed, or were established, during the PLRS acquisition and had some applicability. A short discussion follows.

1. DoD-STD-1467(AR)

Department of Defense Standard 1467, dated 18 January 1985 and titled "Software Support Environment," was originally written as an Army document and later approved for use by all DoD agencies. It establishes uniform minimum requirements for the contractor to define a Developmental Software Support Environment, and ensures the existence of a complete contracting activity life cycle software support capability for the deliverable software from a weapon system procurement. This standard, while recognizing the constraints of the current life cycle software support activities, allows the contractor the flexibility to develop the software and manage the contract in accordance with the contractor's best judgment and practices. [Ref. 13]

2. DoD-STD-1679A

Department of Defense Standard 1679A, dated 22 October 1983 and titled "Military Standard Software Development" was the predecessor to DoD-STD-2167. This standard provided the original requirements for DoD mission critical software development applicable to government contracts. It was concerned with the criticality of software performance and included combat survivability, changing operational requirements (to allow for efficient change of the software), and life cycle costs (with particular emphasis on software design that would reduce these costs). Although this standard has been superseded, it was the software development standard in effect in September 1986. Its requirements were applicable to PLRS. [Ref. 14]

3. DoD-STD-2167A

Department of Defense Standard 2167A, dated 29 February 1988 and titled "Defense System Software Development," establishes uniform requirements for software development applicable to the system life cycle. DoD-STD-2167A was an update to MIL-STD-2167, which had been a major revision of MIL-STD-1679A. This standard provides the basis for government insight into a contractor's software development, testing and evaluation efforts. The standard's intent is not to encourage or discourage any contractor development method, but rather to let the contractor propose a software development strategy that will ensure the best support to meet the requirements of the contract. This standard is tailored by the program manager to allow cost effective measures to be cited in solicitations for contracts. It establishes many of the requirements to be met during the development of the software.

This standard applies to deliverable software components designated as Computer Software Configuration Items (CSCIs) in a contract. All the CSCIs together make up a weapon system's software. The CSCIs are composed of Computer Software Components (CSC), which are, in turn, decomposed into Computer Software Units (CSU). A CSU is the lowest level of software decomposition. While each CSCIs is managed and developed individually, their development follows a master plan that will have all CSCIs integrated at the appropriate time in the weapon system's schedule. This standard, or portions thereof, also applies to the following:

1. Software developed as part of a system or a hardware configuration item but not explicitly identified as a CSCI.

2. Non-deliverable software used in the development and testing of deliverable software and hardware (such as design and test tools).

3. Deliverable, unmodified, commercially available or reusable software.

4. Commercially available software, government furnished software, and reusable software that is modified and delivered as part of the system. [Ref. 15]

This MIL standard has recently been replaced by DOD-STD-498. The DoD-Std-2167 and 2167A were not contractually required by the PLRS acquisition.

4. DoD-STD-2168

Department of Defense Standard 2168, dated 25 April 1988 and titled "Defense System Software Quality Program," contains requirements for the development, documentation, and implementation of a software quality program. This program includes planning for, and guidance on conducting evaluations of software quality associated documentation and related activities. It also provides for planning and conducting follow-up activities necessary to insure timely and effective resolution of identified problems. This standard is intended to be used with DoD-STD-2167A and with DoD-STD-7935A, the DoD Automated Information Systems (AIS) Documentation Standard. This standard, together with other DoD and military specifications and standards (governing configuration management, specification practices, project reviews and audits, and subcontractor control) provides a means for achieving and maintaining quality in software and its associated documentation.

Its intent was to implement the policies of the DoD Directive 4155.1, "Quality Program," (which are now covered under DoD Directive 5000.1) and provide all of the necessary elements that comprise a comprehensive software quality program. The standard interprets the requirements of MIL-Q-9858, Quality Program Requirements, for software. It also superseded MIL-S-52779A, Software Quality Assurance Program Requirements. The software quality activities described in this standard are meant to be applied during each phase of the software system life cycle. The MIL-S-52779A was applicable to the PLRS procurement. [Ref. 16]

5. MIL-HDBK-782

Military Handbook 782 is dated 29 February 1988 and titled "Software Support Environment Acquisition Implementation Guide to DoD-STD-1467." This handbook helps contractors, government acquisition managers, and life cycle software support activities during acceptance of software support by the government's designated life cycle software support activity. [Ref. 17]

6. MIL-STD-1521B

Military Standard 1521B is dated 4 June 1985 and titled "Technical Reviews and Audits for Systems, Equipment, and Computer Software." This standard lists general and specific requirements that both the government PM and the contractor must accomplish during each phase of a review or audit. Technical reviews and audits are conducted in accordance with the standard or as specified in the contract. The standard can be tailored to the particular system and acquisition strategy.

The criteria under which reviews and audits will be conducted during a particular phase of the software cycle are specifically stated. The standard covers the System Requirements Review (SRR) during the System Requirements phase, a Formal Qualification Review (FQR) in the Software Test and Evaluation phase, and a Production Readiness Review when the production decision is being made. [Ref. 18]

E. THE ACQUISITION PROCESS

This section will focus on key elements of the DoD weapon acquisition process that impact software.

1. Historical Perspective

The development of computer software as a separate activity became commonplace during the 1950s [Ref. 43:p. 5]. Initially scientists and engineers, or the end users, developed their own software and ran it on the computers they used [Ref. 43:p.

5]. Not until the late 1960s was computer science recognized as a separate area from engineering [Ref. 43:p. 23]. Within the computer science arena, software engineering did not become recognized until 1968 [Ref. 19:p. 2-1]. Software engineering is still not broadly implemented throughout industry nor a major focus in most university programs. Because of this, the software domain does not have the wealth of time-tested, widely-held practices that other sciences and engineering disciplines rely upon [Ref. 43:pp. 20-22]. Software engineering suffers from the paradigm of not having a single best approach to solving a problem, but only a series of possible solutions [Ref. 43:p. 23].

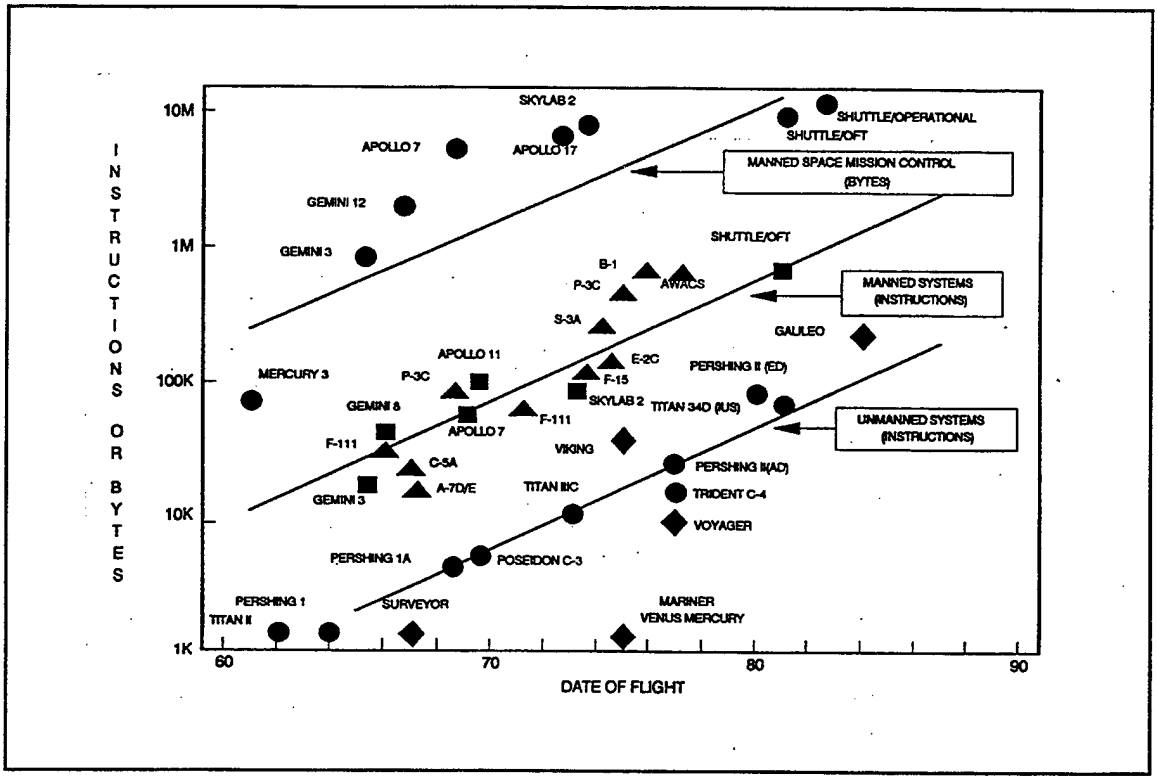
The 1960s saw a rapid increase in the number of digital systems which rely upon computers for their operation. Some of the factors that affected this rapid growth include:

- Advances in integrated circuits. [Ref. 19:p. 2-2]
- Introduction of the microprocessor. [Ref. 19:p. 2-2]
- The Soviet threat forcing the DoD to build fewer, but smarter, and technically superior weapons which relied on computers and software. [Ref. 19:p. 2-2]
- Advances in commercial computers and software brought about by the widespread availability of personal computers and desk-top work-stations. [Ref. 19:p. 2-2] and [Ref. 43:p. 6]
- The realization that software is more flexible and better able to handle change than hardware. [Ref. 19:p. 2-2]

Today all weapons systems, except the most basic, are dependent upon software for their operation. In 1966 the FB-111 required an onboard computer memory of about 60,000 words [Ref. 19:p. 7-1]. By 1988 systems for the B-1B

bomber had approximately a 2.5 million word computer memory requirement [Ref. 19:p. 7-1]. Future systems will exceed these memory requirements. As computer processing power and memory becomes ever larger and less costly per unit, large scale software intensive systems are becoming the norm. [Ref. 19]

Table I. Trends in Software



Source: Mission Critical Computer Resources Management Guide. [p. 7-2]

Table I illustrates the increasing growth and subsequent reliance on software in major systems.

The importance of software over hardware has increased significantly in the last 40 years. In 1950 software had no influence over weapons system design. The

Secretary of Defense's 1990 "Report of the Defense Management Review" cited that by 1980 software averaged from fifty percent to as much as seventy percent of an entire systems cost. Based on this, software has become a critical factor in the acquisition of any weapon system. The annual DoD software expenditure level for MCCR systems grew from about nine billion dollars in 1985 to what is estimated to be over thirty billion dollars by 1990. This continued growth in the quantity and complexity of today's software has strained the DoD's ability to manage the development of software effectively. [Ref. 20]

2. Software Development Life Cycle

DoD-STD-2167A, Defense System Software Development provides guidance on the development and integration of software and hardware in weapon systems. This standard requires a software development effort to include the following major activities:

- System requirements analysis and design;
- Software requirements analysis;
- Preliminary software design;
- Detailed software design;
- Coding and computer software unit (CSU) testing;
- Computer software component (CSC) integration and testing;
- Computer software configuration item (CSCI) testing;
- System integration and testing of the software with the hardware.

During each one of these activities items are evaluated using one or more of these seven criteria:

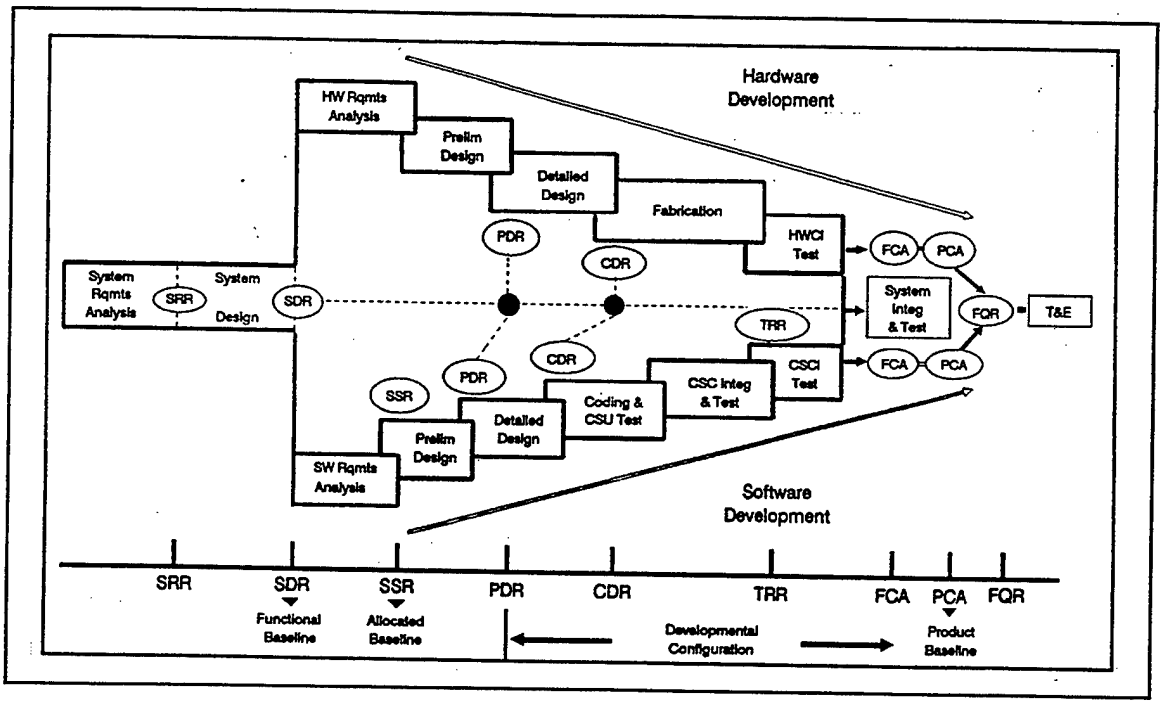
- Internal consistency, to eliminate contradictions in terms and meanings;
- Understandability, to ensure rules of punctuation, capitalization and definition of terms are followed;
- Traceability to ensure a document is in agreement with any preceding documents;
- Consistency between documents that are not related;
- Appropriate analysis, design and coding techniques are used, as stipulated in the contract and the software development plan (SDP);
- Appropriate allocation of sizing and timing resources are not exceeding the documented constraints applicable;
- Adequate test coverage of requirements is addressed by at least one test. [Ref. 15:Appendix D]

Table II illustrates the joint development activities for software and hardware from DOD-STD-2167A. The software development process is done in parallel with the overall hardware development process and integrated in the later stages of acquisition. The following sections provide a short discussion of the software development process.

a. System Requirements Analysis and Design

The main emphasis of this initial phase of the system life cycle requires the government program manager and contractor to: define the overall project objectives; determine the project feasibility; develop an acquisition development strategy; establish a resource cost schedule for hardware, software and personnel;

Table II. Software/Hardware Development



Source: Mission Critical Computer Resources Management Guide. [p. 5-2]

define the interrelationships between hardware and software; and define the technical and business functions as well as the performance of the system. [Ref. 19]

b. Software Requirements Analysis

This is the first phase in the software development cycle. The purpose of this phase, in accordance with the Mission Critical Computer Resources Management Guide, is to establish the functional, performance, interface, and qualifications requirements for each CSCI. The requirements analysis task is a process of discovery, refinement, modeling, and specification [Ref. 43:p.173]. The prototype versions of user interfaces and/or system skeletons may be designed and coded. The developer identifies the support tools and resources, as well as establishes software timing and sizing estimates. The program manager must ensure that all software requirements are traceable to the system specifications. The software development

plan (SDP) is updated and may include the contractors proposal regarding requirements analysis, design, and coding techniques to be used [Ref. 15]. The SDP is a management plan from the contractor to the government outlining how the software will be developed [Ref. 56]. The output from this phase is a final version of the software specifications and an updated SDP. [Ref. 19]

c. Preliminary Design

The preliminary design activity determines the overall structure of the software to be built. This stage deals with turning requirements into data and software architecture [Ref. 43:p. 317]. In this stage input and output relationships, with displays and sensors, are refined according to the hardware configuration and software structure. Also, timing and memory constraints for components are established so that software requirements can be designed to function within the hardware constraints. The developer provides a preliminary design that insures the requirements from software specifications can be traced down to the software components of each CSCI. The developer also generates a software test plan (STP) with the proposed system test program and establishes test requirements for software integration and testing. The STP is generated by the contractor to show the government how the software will be tested, the resources and organization developed to test the software, and any strategies used by the contractor to test the software [Ref. 57:pp. 7-21]

The contractor provides a preliminary version of the software design documents and the STP. During this phase, there is a continual process of informal design reviews, inspections and walkthroughs to evaluate the progress and correctness of the design for each software component. Key documents controlling the software design, test plans, and interface design are put under control, through configuration management, to ensure that changes are documented. [Ref. 19:pp. 5-7]

d. Detailed Design

The purpose of the detailed design phase is to define and complete a software design that satisfies the allocated requirements. The detailed design includes a description of the computer processes to be performed as well as detailed descriptions of the data to be processed. This is done by establishing test cases for the input of data, results from the input of the data, and the evaluation of the software based upon this input [Ref. 15:p. 25]. A key point of these tests is to stress the software to the limits of the requirements [Ref. 15:p. 25]. Any exceptions in coding and special conditions of programming will be addressed in the SDP. At the conclusion of this phase, a critical design review (CDR) will be conducted to ensure that the software design satisfies the requirements of both the system level specification and the software development specifications. [Ref. 19]

e. Coding and Computer Software Unit (CSU) Testing

This phase translates the detailed software design into a software program using a programming language. The source code for a software unit is generated, tested, and compiled into a program when it meets the requirements of the detailed.

This phase depends upon unit testing to eliminate errors, prior to the integration and testing of the whole program in the next phase. There are no formal reviews scheduled during the coding and unit testing cycle. [Ref. 19:pp. 5-9] [Ref. 15:p. 27]

f. Computer Software Component (CSC) Integration and Testing

During integration and testing the software units are combined into a software product that meets the system design. The procedure is done by combining a few components at a time until all components have been integrated and tested. The testing is done by the contractor during this phase, and does not require government

approval. Formal testing, involving government approval, is usually done in the next phase. [Ref. 19:pp. 5-10] [Ref. 15:p. 29]

g. Computer Software Configuration Item (CSCI) Testing

After successfully completing the previous phase, the contractor proceeds with the last step of the software development cycle. Formal tests will be performed in accordance with the software test plans and procedures on each CSCI. Formal testing involves the government verification of software requirements and interface specifications. Testing is done to establish the software product baseline. This is accomplished through a physical configuration audit (PCA) and a functional configuration audit (FCA). The PCA is the technical examination of the software product against its design. The PCA may follow or be done in conjunction with the FCA, an examination of the functional characteristics of the CSCIs.

During this phase the contractor evaluates the software test report (STR) for each CSCI, and the source code and design documents. The source code for each CSCI is delivered to the government, if specified. [Ref. 19:pp. 5-9] [Ref. 15:p. 31]

3. System Integration and Testing

The purpose of system integration and testing is to ensure that the developed software works with the system in the environment that it is designed for. When an acceptable formal qualification review (FQR) is completed the system is turned over to the government for use.

At this point all documentation, source and object code, and any other items specified in the contract are delivered to the government. The government assumes configuration control responsibility. The contractor's configuration management of the software ceases once the product baseline is approved then the government takes delivery and assumes responsibility. In addition, if specified in the contract, the contractor may support the government's system integration and testing, post test

analysis and test results, and system integration. [Ref. 19:pp. 5-12] [Ref. 15:pp. 33-34]

4. Test and Evaluation

Test and Evaluation is a continuous process in the system development cycle. A major purpose of test and evaluation is to demonstrate a system's technical capabilities and operational effectiveness. The results provide key information for decisions on whether to commit significant additional resources to a program, advance from one acquisition phase to the next, or to field the system.

The intent of testing is to provide quantitative data for analysis and to minimize the subjective interpretation of the system's performance. This frequently requires the use of special range instrumentation or measurement and simulation. Testing is conducted by two methods: Developmental Test and Evaluation (DT&E), and Operational Test and Evaluation (OT&E). [Ref. 19]

a. Developmental Test and Evaluation (DT&E)

This testing is conducted throughout various phases of the acquisition process. Following the hardware/software development model in DOD-STD-2167A, testing is a continual process for the baselined system until it is fielded. The DT&E is testing that is conducted by the contractor, but could be witnessed by the government, prior to the system being turned over to the government. It is meant to ensure the acquisition, verification, and fielding of an effective and supportable system by: verifying technical progress by the contractor, substantiating technical performance, and certifying readiness for OT&E [Ref. 49:p. 26].

This testing also includes the test and evaluation of the components, subsystems, and related software. Hardware-software integration, and qualification and production acceptance testing are also included. Additional areas tested deal with compatibility and interoperability with existing or planned equipment and systems.

System effects caused by natural and induced environmental conditions are also evaluated. This testing and evaluation uses models, simulations, testbeds, prototypes or full-scale engineering development models of the system. [Ref. 19]

During software development testing is a quality control process [Ref. 57:pp. 8-43]. Testing may be accomplished incrementally, with more and more of a system tested as it is completed. The DT&E is concerned with unit or component testing, vice system testing. This allows debugging to occur in the components prior to assembly as a complete system. [Ref. 58:pp. 8-45]

b. Operational Test and Evaluation (OT&E)

Operational Test and Evaluation is a field test to demonstrate how well a system operates under realistic conditions and meets its operational requirements. This testing is conducted in an environment as operationally realistic as possible using personnel representative of those expected to operate, maintain, and support the system when it is deployed and includes the threat environment represented by potentially hostile forces [Ref. 49:p. 27]. Independent test agencies oversee the testing and evaluate the results while repairs to equipment during OT&E are done organically. The results of this testing and evaluation are provided for decisions on production and fielding a system. [Ref. 19]

5. Independent Verification and Validation (IV&V)

Independent Verification and Validation (IV&V) is the process that ensures software correctly implements a specific function (verification), and that the software built is traceable to customer requirements (validation) by an agency not responsible for developing the product or performing the activity being evaluated [Ref. 43:p. 632]. Verification is CSCI oriented, while validation is system oriented and they are conducted by government representatives (either government employees or government contractor personnel). This process is an effective quality assurance tool to

complement and reinforce the contractor's software engineering process, configuration management and testing functions. The independence of the IV&V personnel from the software developers is an important point to ensure an objective process.

The decision on whether to establish an IV&V process may be based on the risk associated with software development because of the maturity of the contractor's development tools or system integration procedures. High risk tools and integration procedures may warrant IV&V. Another consideration is whether an error in the software could result in death, injury, mission failure, or equipment loss. In such circumstances IV&V is likely warranted. The IV&V effort is conducted throughout the entire software life cycle. [Ref. 19]

6. Post Deployment Software Support (PDSS)

More than two thirds of the Department of Defense's expenditure for software is for PDSS, commonly referred to as software maintenance [Ref. 19:p. 7-1]. In 1984 the Joint Logistics Commanders defined PDSS as:

> *the sum of all activities required to ensure that, during the production/deployment phase of a mission critical computer system's life, the implemented and fielded software/system continues to support its original operational mission and subsequent mission modifications and production improvement efforts. [Ref. 19:p. 7-5]*

In essence, software is modified to correct a problem or to add a capability or an enhancement that did not exist before.

Three categories of software maintenance were identified by E. B. Swanson in the article *The Dimensions of Maintenance:* corrective maintenance, adaptive maintenance, and perfective maintenance [Ref. 43:p. 689]. Corrective maintenance is performed to identify and correct software errors. Adaptive maintenance adapts software to changes in the environments in which the software is to operate.

Perfective maintenance enhances the performance or improves the capabilities of software. [Ref. 19:p. 7-6]

Improvements in the PDSS process may be provided through use of the computer resources life cycle management plan (CRLCMP). This document is first developed early in the acquisition cycle. It ensures all relevant PDSS issues are properly cited and accounted for. Within the Department of the Navy the requirement for a CRLCMP is spelled out in OPNAVINST 5200.28. The CRLCMP is approved prior to full-scale development and is updated whenever the software is modified. [Ref. 19]

7. Future Trends in the Acquisition Process

The Defense Management Report to the President by the Secretary of Defense in July 1989 provided a framework for future defense acquisition trends. It set forth a plan to implement the Packard commission's recommendations, improve the performance of the defense acquisition system, and manage more effectively the DoD and defense resources. The Report addresses actions to be taken in four areas: personnel and organization, defense planning, acquisition practices and procedures, and government-industry accountability. This report has resulted in significant changes to the acquisition process, some of which have not been fully implemented. More recent severe procurement budget cut-backs have also forced changes in our procurement process. One can expect this domain to continually evolve over time. [Ref. 20]

F. SOFTWARE CHALLENGES

A variety of software problems in government acquisitions have been identified in recent years through GAO reports. While these reports are usually quite general, they nonetheless usually point out major problems. These problems have

been identified as falling into three broad categories: management, requirements definition, and testing [Refs. 21 and 22].

Management problems are those that management had direct control over. They involve cases of managers allowing acquisitions to proceed to new phases before insuring that software development was adequate to proceed, or after software problems had been identified but not fixed. Regarding requirements definition, the problems include: ill-defined requirements and changing requirements. Many systems were not flexible enough to adapt to changing requirements. These problems are one of the biggest factors to over-cost and over-budget software development efforts. Testing was the third area that was identified by the GAO as being deficient. Testing problems included use of inaccurate models and omitting system-level integration testing. The issues and problems faced by PLRS are not unique; most weapon developments have encountered major software challenges. [Ref. 21]

G. CONCLUSIONS

Most of the major MCCR programs require hundreds of thousands, and sometimes millions, of lines of software code to operate. This results in systems that are highly complex. Many must also operate in real time, adding additional complexity and challenges. The process to produce high-quality software for such weapon systems appears to be poorly understood by much of the DoD senior management [Ref. 57:p. 8-1]. The environment in which the development of software is to take place is clearly defined by the regulations within the DoD. Unfortunately, through misunderstanding or neglect, the process is often not followed. This results in added expense, extended schedules, and MCCR that is fielded with reduced requirements. Established software engineering processes and capable tools are

available to develop quality functional software on time and within budget. Anticipated future trends of limited resources and increased oversight, dictate that such processes and tools be utilized if software is to be acquired and fielded properly.

With the publishing of DoD Instruction 5000.2, the number of instructions governing acquisition management policies and procedures are being cut down. Future trends may dictate that more instructions and directives concerning acquisition be put into a usable and easily accessible form, and others consolidated and streamlined within the acquisition arena.

This chapter has reviewed many of the pertinent regulations, instructions and standards which govern MCCR. Although the impact of recent guidance by the Secretary of Defense requiring commercial standards and severely restricting the use of DOD standards is not yet known, this chapter should present a better understanding of the complexity and challenges facing software development for weapon systems.

III. PLRS AND EPLRS HISTORY AND BACKGROUND

This chapter describes the PLRS and EPLRS program history showing where the programs diverged in the 1980s. It describes the capabilities and components of PLRS and EPLRS. It also provides a brief overview of the two system's missions and employment in a tactical environment. Finally, it will discuss the system hardware, software, and firmware for each system.

In the 1980s a variety of DoD weapon systems, including extensive embedded computer systems, were acquired by DoD. The acquisition of the computer resources for these systems was primarily governed by the Warner Amendment of 1981. This act excludes MCCR that perform intelligence activities, cryptoanalytic activities, command and control of military forces, are an integral part of a weapon system, or are critical to the direct fulfillment of military or intelligence missions from many restrictive government acquisition regulations and laws that apply to automated data processing equipment (ADPE). PLRS was excluded under the Warner Amendment.

PLRS has both command and control and communications capabilities. The Marine Corps initiated the PLRS development, and in the mid-1970s was joined by the Army. A joint development and acquisition of PLRS was then pursued. In the mid-1980s the Army identified additional requirements not included in the PLRS program. They then initiated a new development, termed the Enhanced Position Location Reporting System (EPLRS). EPLRS included many PLRS components, including much of the PLRS software.

The historical information for this section has been gathered primarily from a series of government and contractor publications. These include: the July 1990 edition of the United States Marine Corps Tactical Communications Architecture [Ref. 23]; Hughes Aircraft Corporations (HAC) June 1989 EPLRS Briefing Book

[Ref. 24] ; the July 1989 HAC PLRS Briefing Book [Ref. 25]; and the March 1993 HAC brief for the USA Program Executive Officer for Communications [Ref. 26].

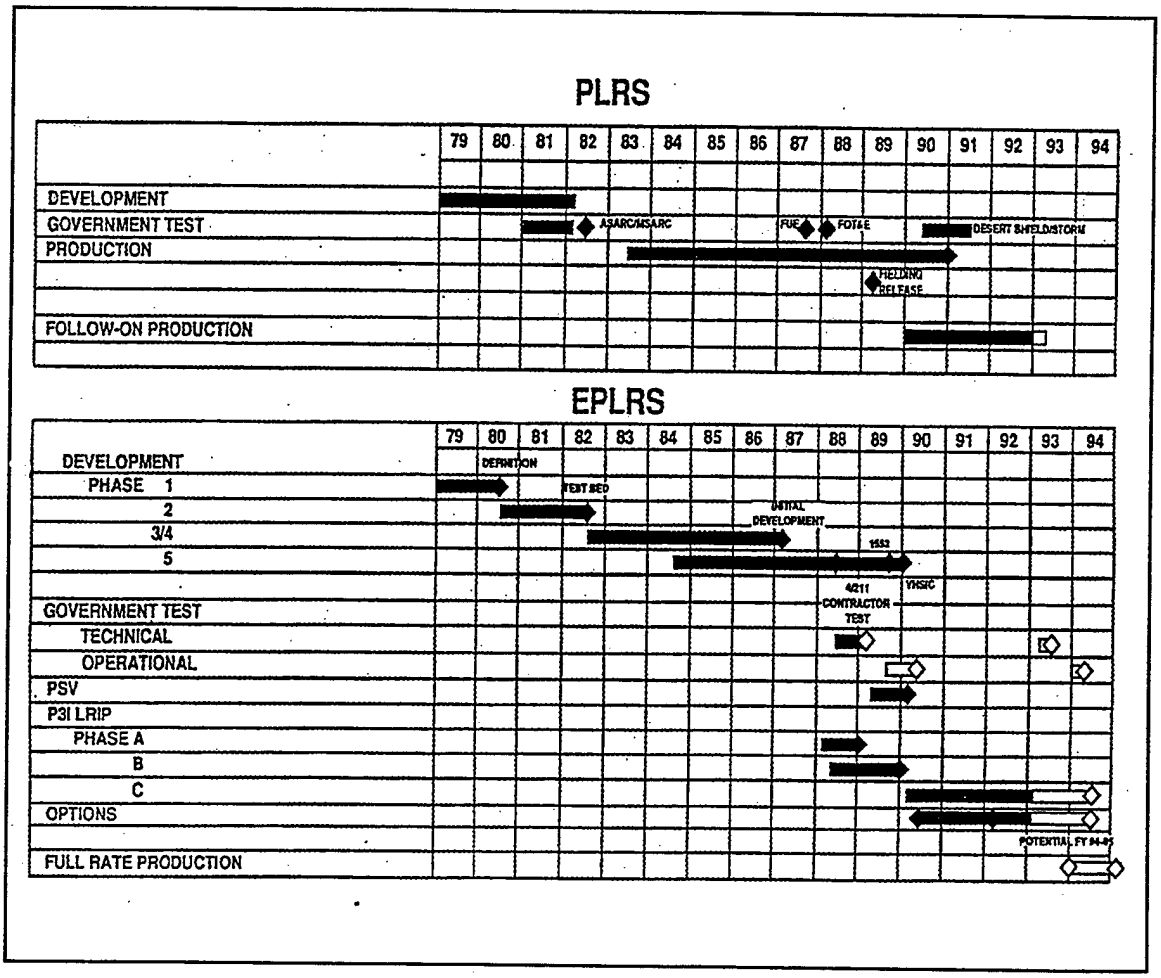
A. PROGRAM HISTORY

The PLRS and EPLRS programs have both been developed by HAC. The initial Specific Operational Requirement (SOR) was originated in the early 1970s from the Marine Corps. The initial Request For Proposal (RFP) called for development of advanced prototype models. Two companies, HAC and General Dynamics, won the initial prototype competition contract. The field evaluations of the advanced development models, or "shoot off," resulted in HAC receiving a development contract for a test model system that included two master stations and 64 user units. These were initially tested and evaluated during 1977 and 1978. [Ref. 26]

During this same time frame a Joint Specific Operational Requirement (JSOR) was approved by the Marine Corps and Army. It resulted in the Army becoming a partner in the PLRS development program and taking on the role as lead service. As such, the Army became the PLRS program manager (PM). A joint program management office was established at the Army's Communications/Electronics Command (CECOM) at Fort Monmouth, New Jersey. This office managed the PLRS development and contract with HAC.

In August 1983, HAC was awarded a sole source 260 million dollar multiyear, fixed-price, incentive firm target contract to produce 23 master stations and 3500 user units for the Army and Marine Corps [Ref. 27]. During the production of PLRS the Army identified expanded requirements for PLRS. In 1987 the Army decided not to field PLRS, but to develop the Army Data Distribution System (ADDS), which uses an Enhanced PLRS (EPLRS) as its major system component [Ref. 27]. The Army then developed a new acquisition strategy. It used the hardware from PLRS, much of which was common to EPLRS, for the initial EPLRS units. The PLRS software

was largely used as a baseline, with additional functionality added for EPLRS. Figure 1 depicts the timelines for major phases of both programs as they progressed from 1979 to the present. Because of the similarity of both programs, even though the Army had additional requirements, the joint PM remained at CECOM. [Ref. 28]



Source: HAC brief to BGen Gust, United States Army on PLRS/EPLRS, 3 March 1993.

Figure 1. PLRS/EPLRS Program Development

The Marine Corps managed the PLRS program through a composite group named the Acquisition Coordinating Group (ACG). The ACG included representatives from Headquarters Marine Corps, the Marine Corps Research and Development Command, Marine Corps Tactical Systems Support Activity and the PM office at CECOM. The ACG was concerned with the development, budgeting, logistics, training, personnel staffing, testing and fielding of the system. It oversaw the PLRS procurement for the USMC and took action on problems related to: the work performance of the contractor (through the PM office); funding between the Army and Marine Corps on research, development and production; post-deployment software support issues; as well as any other issues that required a USMC decision for the PLRS program. The ACG met monthly in Washington, D.C. to review the program, discuss problems and take action for resolving them. With the restructuring that resulted from the activation of the Marine Corps Research Development and Acquisition Command (MCRDAC) the ACG was later replaced by a linear decision authority that consisted of a program manager overseeing all ground communications systems and an assistant program manager and staff that specifically managed PLRS. It relied on passing a problem, recommendation or solution through a series of responsible offices. Decision authority started at MCRDAC. [Ref. 28]

As PLRS was in production, EPLRS was undergoing development. PLRS was fielded to Marine Corps units in 1987. In 1990 PLRS was deployed with Marine Corps units to Saudi Arabia and supported Operations Desert Shield and Desert Storm.

In 1988 EPLRS prototypes began a series of technical and operational tests conducted by the government. Some significant deficiencies, both hardware and software, were discovered during these tests. Presently, EPLRS is going through a

new series of testing by the government as part of the Pre-Planned Product Improvement (P3I) Low Rate Initial Production (LRIP). [Ref. 26]

B. MISSION

The primary mission of PLRS, as spelled out in Developmental Bulletin 1-87, is to give the user navigational assistance. The PLRS navigational information is also available to the force commander and staff to control ground units and coordinate their supporting arms. PLRS is designed to be highly resistant to electronic warfare threats and can provide a limited digital communications capability to users.

PLRS provides a communications network and was designed to support a Marine Expeditionary Brigade (MEB). Multiple systems can be employed to support larger or smaller task organized units as a commander's need dictates. A single PLRS consists of two Master Stations (MS), a maximum of 370 active User Units (UU), and a maintenance van. [Ref. 29]

This system supports the Marine Corps' reliance on single channel radios during the assault phase of amphibious operations and high mobility operations once ashore. PLRS does not rely on single channel radios and can perform its mission when the reliance on communications transfers to a more permanent switched backbone system. [Ref. 23]

EPLRS makes up one element of the Army's plan to integrate their communications capabilities on the battlefield. It is to be employed at the Corps level. This system is comprised of three major components: Mobile Subscriber Equipment (MSE) (the phone system), Combat Net Radio (CNR) (voice radio nets), and the Army Data Distribution System (ADDS), of which EPLRS is the major component. EPLRS, because of its added digital message capabilities, is expected to cut the traffic load on voice radio nets by 40 percent or more. The following paragraphs provide a

brief overview on the tactical employment of PLRS and EPLRS as well as their major hardware and software components.

C. TACTICAL EMPLOYMENT

The primary purpose of PLRS is to serve the location needs of users assigned a UU and provide location information to the commander of the equiped units. In addition to this, EPLRS provides enhanced digital communications capabilities. This section briefly discusses these functions.

1. PLRS

The PLRS MS has the ability to give real-time, three dimensional data on any unit controlled by that MS. Users equipped with a UU also have the ability to access that data. The User Readout (URO) of the UU provides user position, position of other UU's and their military identification, and the range and bearing to other units.

Additionally, navigation information can be provided to a UU on range and bearing to Pre-Designated Items (PDI). These include such items as bridges, road junctions, checkpoints or objectives. Lanes and zones can be designated to aid landing craft in reaching the correct point on the beach or identifying restricted zones, such as mine fields or impact areas. Corridors, with specified upper and lower altitude limits, can also be designated to guide airborne users to their destinations while avoiding danger zones. [Ref. 29]

2. EPLRS

The Army has retained the basic requirements provided in the Marine Corps PLRS system and added requirements to support air defense artillery, fire support, combat service support, maneuver, and intelligence/electronic warfare. Within these areas the intent is to reduce voice communications through the digital data communications capability provided by EPLRS. This takes the form of fire mission requests for artillery, air track from the sensor to the Forward Area Air Defense (FAAD) units,

and data provided from the forward sensors to the All Source Analysis System for the Intelligence/Electronic Warfare (IEW) area. [Ref. 24]

D. SYSTEM HARDWARE

1. PLRS Components

There are three major hardware components for PLRS, the master station (MS), user unit (UU), and direct support team vehicle (DSTV). The MS provides the central network management for the system. It performs a variety of functions that include: processing of position and navigation information to users, identification of each user of the system, limited digital communications traffic between users, and the location of the user units within the coverage area. The master station is contained in a S-280 shelter and is transported by a five-ton truck or by helicopter. [Ref. 29]

The MS contains three mainframe computers, one UYK-7 and two AN/UYK-44s. In addition, the MS has peripheral equipment with embedded processors. These include: the Display Control Console (DCC), the Forward Area Teletypewriter (AN/UGC-74), and the Command Response Unit (CRU). The original requirements were for two AN/UYK-20 computers, the standard Navy tactical computer at the time of procurement. The first MSs delivered contained the older generation AN/UYK-20s which were later replaced by the follow-on generation AN/UYK-44 computers. The AN/UYK-44 is a militarized reconfigurable computer and was designated in TADSTAND B as the successor to the AN/UYK-20. In a November 1985 Statement of Work (SOW) change, HAC was tasked to change the Government Furnished Property (GFP) in the MS to the AN/UYK-44 computer. The SOW required HAC to initiate engineering efforts, that included mechanical design, software design, and documentation, for conversion to the new computer. [Ref. 30]

The UU is a solid state, militarized transceiver. It receives and responds to MS generated command messages. The UUs are individually identifiable to the MS; they

receive, transmit, relay, perform range measurement, and perform message processing functions necessary for position location and communications within the PLRS network system. Unit identification and position determination are done automatically at the UU by the MS. [Ref. 29]

The UU is controlled by a low-power microprocessor, the signal message processor (SMP). It accepts, interprets, and acts on commands generated by the MS. The SMP schedules the unit to transmit, receive, acquire network synchronization, measure signal time of arrival, send status, relay data, originate or receive data messages. It controls the turn-on/turn-off of internal circuitry to minimize power drain on the unit. The UU also has an internal control that measures atmospheric pressure for altitude determination. The MS converts this pressure measurement into an elevation and sends it back to the user unit. [Ref. 29]

The UU is implemented in several configurations which include: manpack, vehicle, airborne unit, or as an auxiliary ground unit. The manpack version has a receiver-transmitter and URO carried on a pack frame. The vehicle configuration has a mounting kit for attaching the radio to a vehicle and its electrical system. The airborne unit is mounted to an aircraft and includes a pilot control display panel in the aircraft crew compartment. The airborne unit is currently installed in some Air Force F-16s. Future plans call for installation into the Marine Corps AV-8, Air Force A-10, Army UH-1 and CH-46 helicopters [Ref. 58]. [Ref 29]

The UU SMP relies upon Erasable Programmable Read Only Memory (EPROM) to control its functions within the network. Messages and displays are shown on a User Read Out (URO) device which is a handheld component of the UU. The URO can display 22 alphanumeric characters and messages of up to 10 characters can be sent and received. It also has two status indicators. The status indicator lights indicate that a message has been received or that the UU is off of the network. The

ser read out (URO) also houses an output device consisting of a 16-key matrix with six function keys controlling the keyboard display operation modes and ten keys controlling alphanumeric entries. [Ref. 31]

The third major PLRS component, the DSTV, is contained in an S-280 shelter and is transported by a five-ton flatbed truck. It is equipped with the facilities to conduct maintenance on PLRS equipment. There are adequate facilities to conduct second through fourth echelon maintenance on all PLRS components in the DSTV. The PLRS Test Set (PTS), a piece of special test equipment, is used in the DSTV. The PTS contains circuitry and programs that simulate the PLRS network and MS signals. It is used to conduct readiness tests on the URO, UU, and the command response or signal processor of the MS. [Ref. 29]

The PLRS system, as fielded, operates in the ultra high frequency (UHF) range, 420 to 450 megahertz, using a Time Division Multiple Access (TDMA) multiplexing technique. This provides flexible traffic loading on the network and more efficient bandwidth usage [Ref. 59] Electronic counter-counter measure (ECCM) is accomplished using spread spectrum, frequency hopping, error detection/ correction, and time slot scrambling. Communications Security (COMSEC) is provided through an internal secure data unit. The network of PLRS stations is centrally controlled from the MS. [Ref. 29]

2. EPLRS Components

EPLRS includes an Enhanced PLRS User Unit (EPUU) and a URO module, a Net Control Station, (similiar to the PLRS MS), and an EPLRS test set. Currently, the equipment for EPLRS consists of many of the same internal components as PLRS.

The EPUU has an expanded message capability. A key purpose of EPLRS is to reduce voice communications and add secure digital communications to the battlefield [Ref. 24]. The SMP for PLRS was redesigned for EPLRS to accommodate

the expanded communications requirements and capabilities. The EPUU also provides a capability for users in the same area to establish a local subnet and communicate with each other without going through the NCS. This local communications subnet provides a capability to relay messages to other members of the group through the members of that net. [Ref. 24]

The NCS is currently housed in an S-280 shelter and is carried on a five-ton truck. The Army is currently attempting to downsize it to an S-250 shelter for use on a HMMWV. This will enable it to use less power and be more mobile on the battlefield.

The EPLRS test set was derived directly from the PLRS test set. It will provide the same maintenance capabilities that the PTS does for PLRS. [Ref. 24]

E. SYSTEM SOFTWARE AND FIRMWARE

The set of instructions that control the PLRS and EPLRS systems are divided into two distinct categories: software and firmware. Software, as defined by the Federal Acquisition Regulations, is the set of instructions and data that are executed in a computer [Ref. 19:p. 3-5]. The computer software enables the computer to perform computational or control functions.

The combination of hardware and software on a computer chip is termed firmware. Firmware is defined as software that has been implemented in hardware using memory devices such as read only memory (ROM), programmable ROM (PROM), erasable PROM (EPROM), or electrically erasable PROM devices (EEPROM) [Ref: 19:p. 3-6]. The job of firmware is to interpret the control functions of the equipment. PLRS uses EPROMs.

There are three software programs that control the MS's functions: the Time of Arrival Program (TOAP), the Network Control Program (NCP), and the Database Control Program (DCP). Each of these is programmed in the CMS-2 computer

language. These three computer programs control the operation of the MS and together are called the Real Time PLRS Program (RTPLRS).

The firmware for PLRS is incorporated into the UU. It is divided into URO firmware, UU firmware, Command Response Unit (CRU) firmware, and the Display Control Station (DCS) firmware. The firmware and software for these components are all programmed in either Assembly, Microforth, or a combination of these two programming languages.

Additional computer hardware, software and firmware are used to support the MS trainer, the software test tools, the PLRS test set, program test sets, and other support functions. These use a variety of computer programming languages. Future trends envision software upgrades for PLRS and the support functions including transition to the use of the Ada programming language [Ref. 32]. Currently, EPLRS is using CMS-2 to control the MS functions. Appendix E provides the size for each of the software and firmware programs for PLRS.

F. SUMMARY

This chapter has provided a brief overview of the technical characteristics and operational capabilities of PLRS and EPLRS. Both programs began with the intent to produce a command and control capability for the commander on the battlefield. The Marine Corps developed and fielded a system with these capabilities. The Army expanded the system requirements to include a greatly expanded data communications capability, making EPLRS into a key component of the Army communications architecture. EPLRS was fielded in fiscal year 1995.

IV. PLRS SOFTWARE AND FIRMWARE ISSUES

The focus of this chapter is on key software, firmware and hardware problems that were identified during the development, testing, software delivery, and early post deployment software support (PDSS) of the PLRS program. It examines the product delivered to the government, the acquisition philosophy that was used, and PDSS.

A. SOFTWARE DEVELOPMENT

Traditional DoD software development consists of several areas addressed in Chapter II and depicted graphically in Table II. This section does not follow the PLRS software development through each one of the phases. Instead, it will focus on system integration and the physical configuration audit that occurred before the system went to testing and evaluation. It also looks at the EPROM utilization in the UU and the requirements that were changed to accommodate unexpected growth in the program.

1. System Integration Testing

System Integration Testing addresses both software and hardware performance. It is accomplished through a Functional Configuration Audit (FCA) and a Physical Configuration Audit (PCA). [Ref. 19:pp. 5-12] The PCA is addressed because of the significant impact it had on the PLRS production program. Because the PCA provided a detailed look at the supportability of PLRS software, it served as an indicator of the general health of the software. It also categorized the problems identified and how they would affect the support of the software after delivery to the government. This was very important to MCTSSA because it was preparing to assume the PDSS functions from HAC.

At the request of MCTSSA a partial software PCA was conducted in May 1987, as part of the formal qualification review. An FQR is a system-level review

that verifies that actual system performance complies with system requirements [Ref. 19:pp. 5-12]. MCTSSA had requested the review because no software or firmware audit had previously occurred. This audit was done on the Database Control Processor (DCP), Network Control Processor (NCP), and Time of Arrival Processor (TOAP). The PCA was divided between representatives from CECOM and MCTSSA. The MCTSSA audit team did the audit on the DCP and CECOM did the audit on the other two programs. The initial audit by MCTSSA was done on ten percent of the DCP program. The audit found 101 general software deficiencies. On the basis of the large number of discrepancies discovered in the initial audit, a decision was made to complete a 100 percent audit of the DCP software. The Software Quality Checklist from the PCA is attached as Appendix A.

The audit was conducted to define a production baseline for the DCP as a part of the RTPLRS software. The software production baseline is applicable to the production and deployment of a system. It contains the objectives for key cost, schedule and performance limits [Ref. 3:p. 11-A-1]. It also provides a means to examine the contractor's compliance with the requirements of the contract and progress toward delivery of a system [Ref. 19:p. 10-6]. A PCA is done by determining if supporting documentation reflects the system as it was built and as specified in the program performance specification (PPS). Ideally, an audit would show that the documentation contained the requirements. A key purpose of the PCA was to determine whether all materials that were provided met the needs of the PDSS activity. This would ensure the life cycle support of the program could be effectively accomplished. [Ref. 33]

Because the PLRS production contract was awarded in 1983, the software standard under which the audit was conducted was DoD-STD-1679A. By contract the software was only required to be in accordance with paragraph 5.9, Software

Quality Assurance, and paragraph 5.11, Software Configuration Management, of this standard. The Software Quality Assurance section of this standard requires the contractor to implement software quality assurance policies, practices and procedures to verify that the product will meet the software requirements approved by the contracting activity [Ref. 14]. This section requires independent software quality audits that measure system conformance. The Software Configuration Management section requires the contractor to implement policies, practices and procedures to ensure the positive identification, control and status accounting of the software and all items of deliverable software documentation during all phases of the development effort [Ref. 14]. The PLRS Engineering Design Model (EDM) documentation contained the remaining standards to audit the software. The PLRS EDM was a prototype of the system developed prior to production. Unfortunately, the EDM documentation was not supplied to the audit team for the audit.

The MCTSSA audit found deficiencies in several areas including: external documentation, software code internal documentation, naming conventions, code structure simplicity, machine independence, modularity, and error checking (Appendix B contains the software quality checklist used for the partial physical configuration audit). Category A discrepancies were of particular concern. These are deficiencies that show a lack of information that the developing contractor must provide before the PDSSA can assume the maintenance functions on the software. Failure to provide this information would make maintenance by a PDSS activity extremely difficult, if not impossible. The major category A discrepancies for PLRS included the following:

- Lack of system architectural information.

- Information relating to the design structure was scattered throughout the documentation and code listings.

- Interface design specifications between the NCP, TOAP, and DCP were only minimally documented.

The audit team for the DCP recommended to the government PM that all category A items be fixed before acceptance of the software by the government. Additionally, it recommended that the software development and procedures in DoD-STD 1679 that were not required under the existing contract be noted and included in future developments. A number of these had been practices that were established but not documented in the EDM. [Ref. 33]

Meetings occurred in July 1987 between the contractor and the audit team to discuss the problems associated with the PCA. These meetings discussed specific answers to the deficiencies and attempted to resolve them. [Ref. 34] All the identified category A deficiencies were resolved to the goverment's satisfaction by HAC before delivery of the software to the Marine Corps in 1987.

2. Memory Availability

Memory availability is a major concern for any fielded computerized system. Reserve memory supports any future planned or unplanned software and firmware upgrades by enhancing or providing new capabilities to a fielded system. Memory requirements generally reside in three areas: main memory, secondary storage, and support software. Main memory consists of the components of the system where software programs are executed and within which data is stored [Ref. 11]. Secondary storage is that component of the system that is used as an auxiliary to main memory and is achieved by bulk storage or magnetic tapes and disks [Ref.11]. Support software is that software, firmware, and data that are the means by which the system

is developed, tested and maintained. It includes software supporting simulation, configuration management, and utility programs for various activities such as compiling and debugging [Ref. 11]. The firmware is limited in memory size by the capacity of the read-only-memory (ROM) hardware device that it is loaded on [Ref 57:pp. 2-7].

The SMP in the UU contains a microprocessor and a stored software program, loaded in firmware, to run the functions of the UU. The memory capacity of the UU is divided into two parts: the Ultraviolet Erasable/Programmable Read Only Memory or EPROM (16 kilobytes) and the Random Access Memory or RAM (1 kilobyte). At the conclusion of development and testing, the memory storage capacity of the SMP was nearly depleted. Under the production contract this was acceptable. Since the firmware software could not be expanded without replacement of the EPROM, future growth or changes in the system software would be severely limited. [Ref. 31]

A HAC recommended solution to the memory constraint was to expand the SMP to accommodate 32 kilobits of total EPROM memory and 8 kilobits of RAM. The recommended SMP upgrade would also have the capability to expand an additional 48 kilobits for future expansion. [Ref. 31] This upgrade was included in a 1989 ROC for enhancements to PLRS, which was approved and is currently under contract [Ref. 35]. This will require all SMPs in the UUs be changed.

Additionally, there was very little room for growth within the UYK-7 and UYK-20 MS computer's memory. The NCP and DCP functions were executed on the UYK-20s. The initial MSs were configured with UYK-20s and had serious limitations. These computers had a standard 256 kilobits of RAM. Because of memory restrictions, programs had to be swapped in and out of memory. Which is time consuming and made the system inefficient. While there were memory constraints in the UYK-7s, the TOAP used a stable algorithm that did not require use of memory

swapping and therefore was not subjected to the limitations of the NCP and DCP. The prime reason for replacing the UYK-20s with UYK-44s was to give the system increased memory capacity. [Ref. 36] As previously mentioned, the decision for the UYK-44 upgrade was made after the first MSs were finished and required the initial delivered production units to be retrofitted. This conversion was required to comply with TADSTAND B. Besides the hardware conversion, the software had to be modified to run on the UYK-44 computer [Ref. 9].

B. SOFTWARE TEST AND EVALUATION

A good indication of the quality and maturity of a software/firmware system can be shown by testing it in an environment that is similar to the one it can be expected to operate in. Problems that may have been identified during development can be tested and evaluated against the ROC. If successfull testing and evaluation can put any program closer to fielding; if unsuccessfull it can indefinately postpone fielding until problems are fixed.

This section addresses the developmental and operational testing and independent verification and validation associated with the test and evaluation of PLRS software. It also includes discussion on the in-process review (IPR) used to track the current status of the PLRS program, throughout its development. The IV&V and IPRs were methods used to discover and resolve softrware problems early and thus keep the program on time and within budget.

1. Developmental Testing

Between 1981 and 1986 developmental testing was conducted by HAC and the government. The first three major tests were done with the PLRS EDM; the last test was with production hardware and software.

Government Development Test II/Operational Test II (DT-II/OT-II) was conducted at Fort Huachuca, Fort Hood and Camp Lejeune in 1981 and 1982. A

review was held in 1982 that granted provisional approval for service use and the PLRS program gained a milestone III limited production approval. However, the milestone III approval required the service to conduct a follow-on test and evaluation (FOT&E) with the initial production units. The results would be evaluated prior to any further production approval. From 1982 to 1986 both Marine Corps and Army units continued to use the PLRS EDM in operations in the United States and Europe.

2. Follow on Test and Evaluation (FOT&E)

In November 1986, because of the Army's decision to field EPLRS, the Marine Corps assumed responsibility for the PLRS follow-on operational test and evaluation (FOT&E). A FOT&E is used to reevaluate the system to ensure that it meets operational needs and operates in a new environment or against a new threat [Ref. 56]. The follow-on operational testing and evaluation for PLRS was conducted over a two week period in February and March 1988 at the Marine Corps Air Ground Combat Center, Twentynine Palms, California. The conduct of the test and the analysis of the results was the responsibility of the Marine Corps Operational Test and Evaluation Activity (MCOTEA). The first week was a free-play MEB exercise using PLRS. The second week involved scripted play in an electronic warfare environment with a community of 370 Uus. [Ref. 28]

Problems related to a variety of areas were identified in the FOT&E at Twentynine Palms. During the scripted exercise one of the primary mission deficiencies identified was the system's inability to track a full 370-unit community. During the first two days of the test the MS and Alternate MS tracked up to 275 UUs before a mission failure occurred. This was the result of system loading (a software problem) vice hardware component failure. The maximum number of units tracked during the test was 358 UUs. This fell short of the 370 UU requirement. Also vertical position accuracy and message response time for a user community of this

size were not satisfactory. The response time for messages did not meet the stated requirement. An 85 percent response rate within a 45 second service time was recorded. The requirement criteria for success was a 90 percent response rate within 45 seconds. [Ref. 28]

A questionnaire was administered to commanders and executive officers of battalion size and larger units, as well as other unit leaders. There were five areas measured in the questionnaires:

- Did PLRS facilitate location and control of maneuver units?
- Did PLRS affect fire support coordination faster than other means?
- Did PLRS facilitate navigation control?
- Did PLRS affect resupply more efficiently?
- Did PLRS reduce the number of voice transmissions?

Based on the system's performance the respondents did not feel it was an improvement over existing techniques in these areas. Overall PLRS was judged to have failed to meet the criteria improving these areas. [Ref. 28]

It was learned later that the during the exercise some units shut off and moved reference units out of their concern for keeping custody of government property. Because of this a large number of units were disconnected from the network and never able to re-enter again. This lead to a large amount of frustration from units, and was a factor in their answers to the questionnaires. [Ref. 36]

The hardware for PLRS was also evaluated during testing based on the mean-time-to-repair. This was based on repairs to the MS and UU during the test. The MS did not meet the mean-time-to-repair criteria at first and third echelon for the test.

The UU successfully met the requirement at first and third echelon maintenance for the test. [Ref. 28]

The contractor made temporary corrections to the software that corrected the problems related to UU acquisition while the system was still undergoing tests. These were identified as Level 1 faults. They are considered mission critical failures with no way for the operator to work around the failure and fix it. Even after these corrections the system was unable to track the required 370 units to satisfy the criteria of the test [Ref. 36]. An "after the fact" look at the PLRS level 1 faults indicated a weakness in the design of the software, and in particular an input/output chain failure [Ref. 36]. [Ref. 28]

Because of these test results it was recommended by MCOTEA that PLRS not be approved for full-scale production until the major discrepancies were corrected and verified in a future operational test [Ref. 28]. The PM did not recommend additional testing. This was based on the system's ability to support a MEB with a 225 UU community. The PM believed the tests had met the requirements because of the software fixes performed by Hughes Aircraft during testing to correct software problems in the MS. [Ref 36]

Additional doubts of the validity of the test were raised primarily because of the priority and weight that was given to deficiencies identified during the test. For example, should a software problem that was corrected prior to the completion of the test be given the same weight as a failure to adequately train personnel properly in the employment of the system? The credibility of the system evaluation was questioned. [Ref. 36]

3. Independent Verification and Validation (IV&V)

Independent Verification and Validation (IV&V) is a manner for an independent agency, or third party, to evaluate software and associated software products for

compliance with requirements and specifications set forth in a contract [Ref. 57:pp. 9-53]. Because verification is CSCI oriented (refer to IV&V, Chapter 2) it can be completed prior to the complete system integration. However, validation is system oriented (refer to IV&V, Chapter 2) and can only be shown by completion of a DT&E and OT&E. As such, the 1988 OT&E was an integral part of the validation process for PLRS.

During the development of PLRS the software support activity did not have representatives conducting IV&V in-plant at HAC. The program manager office had a resident technical representative (RTR) office at HAC that was responsible for IV&V. The RTR personnel reported to the PM at CECOM vice directly to the PDSSA. In interviews with the PLRS project officer, the PLRS personnel at MCTSSA were not comfortable with this arrangement. They were concerned that their interest in software was not being adequately addressed. However, the software support activity at MCTSSA had been offered the opportunity to place four software personnel in the HAC PLRS software development office, but did not feel it had an adequate number of personnel to do so. Instead it chose to keep its personnel at MCTSSA to conduct checks on the documentation coming in from the contractor. At that time the amount of documentation coming to the PDSSA for review was overwhelming the assigned personnel's ability to check it. In hindsight, it can be said the importance of in plant IV&V representation from the PDSSA was not fully recognized for the PLRS development. [Ref. 36]

4. In-Process Review

During the development of PLRS there was an In-Process Review (IPR) at least every six months. During a period when the contractor was having serious problems the IPRs were held quarterly. These reviews brought together government representatives from MCTSSA, MCRDAC, and the PM office with contractor

personnel for a formal program review. These reviews were not meant for the technical interchange of information, but rather as an update on the progress the contractor had made to meet the government contractual requirements. The matters discussed involved details at a level important to a PM. They did not go to the programmer level of detail. However, major software and hardware issues were covered in their reviews. [Ref. 36]

The reviews provided all parties the opportunity to discuss the testing, timelines, program trouble reports and any other areas dealing with the firmware, software and hardware for PLRS. [Ref. 36] These meetings provided an opportunity for face-to-face discussions on outstanding program issues. Open issues were identified and corrective action assigned.

C. SOFTWARE CONFIGURATION MANAGEMENT TRANSFER

Software configuration management transfer from the development contractor to the government involves the contractor delivering the software, hardware, documentation and other items necessary for the government to provide software life cycle support. Training is also included as a part of this process. This section covers the transition of configuration management and delivery of PLRS software to the government. Some of the transition processes were going on at the same time as the software testing and evaluation.

1. Documentation

Documentation for PLRS consists of two types: documentation for the maintenance support of PLRS software, and documentation for PLRS system operation. The software support activity was concerned with the documentation to support the PLRS software. This documentation was necessary in order to understand how the PLRS software and firmware were coded and designed. The PDSSA was

responsible for upgrading the software and firmware, as well as fixing software errors. Documentation was vital to conducting their mission.

The software documentation was presented in a multi-level manner. The "A" specifications represented the Requirements Operational Capabilities (ROC). The "B5" specifications represented the computer program software level documents. The "C5" specifications represented the product documentation. Additional documentation included the interface requirements specifications (IRS), the engineering notebooks (ENB), and the user manuals for specific items, such as the interactive debugger and software maintenance support facility (SMSF) downloader.

There were 807,000 pages of documentation delivered to the government for PLRS [Ref. 36]. The PDSSA checked the documentation as it was delivered to primarily ensure it was in the correct format, grammatically correct, and that the requirements could be traced through various levels of documentation. MCTSSA's concern was that programmers be able to understand how the code was organized. The code was traced from A level (the highest level) through the B and C levels, much like a data flow diagram traces the flow of data through a system. Every requirement had to be implemented at some point in the code. One purpose of the PCA was to review some of the documentation and ensure it met the requirements for support. However, time and fiscal restraints did not permit a 100 percent review of all the documentation. Samples were used as indicators of the quality and completeness of the documentation.

In checking the documentation some discrepancies were discovered. For example, the description of the PLRS software interface with the system cryptographic function was incomplete. HAC was initially reluctant to talk about the specifics because of security concerns. The National Security Agency (NSA)

eventually agreed that MCTSSA had a need to know about the classified information. To resolve this issue HAC created a classified supplement to the documentation.

During the documentation review process, a somewhat adversarial relationship developed. MCTSSA perceived a reluctance on the part of HAC to voluntarily reveal problems. HAC did not fully agree that all of MCTSSA's requests were within the scope of the contract and that they necessitated additional work. The PM office at CECOM resolved conflicts. [Ref. 36]

2. Training

This section discusses the training that was necessary to get PLRS PDSSA personnel ready to support PLRS. It involves both the PDSSA preparing its personnel for the tasks involved in support, as well as the contractor supporting the transition by providing training to the government's software support activity.

One of the first problems recognized in the transition from the contractor to government support was a lack of training for the personnel at the PDSSA. No contractual agreement existed to transfer the knowledge that the contractor had gained during PLRS software development to the software support personnel at MCTSSA. This unfunded requirement was identified late in the program. MCTSSA received $500,000 from MCRDAC prior to software support transitioning for training. Additional funds were later provided at the expense of other spending. MCTSSA chose to award a contract to UNISYS to develop a training course, deliver training materials and train 20 programmers at MCTSSA. The deliverable training materials would be reused for future training by MCTSSA. The primary purpose of the training was to teach the architecture of the CMS-2 programming language down to the bit stream flow level.

The personnel at MCTSSA also needed training on the PLRS application utility software. Training for the SCENGEN, IDATANA, and data reduction tools

was provided by two companies, Eagle Technologies and Sierra Cybernetics. No arrangement was made for training by HAC for the RTPLRS and ISIMPLRS software. This was due in part to the high price HAC wanted for training, as well as the adversarial relationship that had developed between MCTSSA and HAC.

3. Transition Plans and Process

Four documents covered the transition of PLRS from the contractor to the government: a brief by HAC on the transition process [Ref. 38], NAVMATINST 5200.27 [Ref. 39], the Software Transition Support Plan for PLRS [Ref. 37], and Data Item Description DI-E-7142 [Ref. 40]. Each contained information on the transition of configuration management and the software support facility to the government. The obligations for the contractor and government to ensure a smooth transition were covered in these documents. A brief overview of each document follows.

a. PLRS Software Maintenance Transition and Training Brief

Hughes Aircraft Company provided a brief to the government on the transition process on 21 November 1986 that discussed the software and firmware support facilities, the equipment and approximate transition time frame. Near-term transition concerns of the contractor were also presented. This brief communicated the contractor's understanding of the transition process and their obligations.

The brief showed the connectivity within the program support facility, software development facility, firmware support facility, and display control console firmware support station. This was the equipment HAC used to develop the PLRS software. Each of these represented critical components for the life cycle support process. Appendix C gives an overview of the key components of this Software/ Firmware Maintenance Support Facility (SFMSF). [Ref. 37]

The PLRS program support facility (PSF) is a computer string used for integration and test of PLRS software. The "string" is the term used for a MS that is

unsheltered. The string contains all of the components of a sheltered MS. This set-up is used to run PLRS with simulation scenarios. There are two of these scenarios, the Fulda Gap and Norway. Two strings made up the PSF, representing a master station A and B. These were important for running intercommunity transfer of NCS operations and to test the scenario with an alternate NCS. Within this string there were three important components for running the scenarios: Display Control Console (DCC), Cartridge Magnetic Tape Unit (CMTU), and the Command Response Unit (CRU). Appendix E to the software transition support plan contained the Engineering Notebook (ENB) that detailed the demonstration procedures that verified correct operation of the PSF:

- The DCC is a console that provides the human interface to PLRS. The user inputs commands via keyboard, switch actions or trackball signals. Data is then provided as output to a monochrome display monitor in graphic and tabular form.

- The CMTU uses magnetic tape as a recording medium. It is used to record a scenario.

- The CRU is the radio frequency and signal processing part of the MS. It provides the interface between the NCP and the PLRS network. It is used within the "string" as a simulated community of users.

The strings were also connected to a VAX 11/780 computer through a Rockwell Board which processed the scenario for the strings. This board allowed the string and VAX to talk logically to each other.

The SDF demonstrated the connectivity between the DEC VT100 terminals and the VAX 11/780. These terminals were used to make changes to the source code and compiled the program. Appendix D to the software support transition plan contained the demonstration procedures for the SDF.

The brief also showed a line diagram on how the Firmware Support Facility (FSF) was to be configured. This facility supported the firmware and firmware changes for PLRS. A detailed inventory of all items necessary for the operation of the FSF was provided in the Engineering Notebook (Appendix B of the software transition support plan).

The DCC firmware support station provided a means to test the PROM once it was programmed. It was connected to the PSF through the DCC. The shortage of a DCC firmware support station required a demonstration of this capability to be made at the HAC plant vice at MCTSSA. This was in accordance with the software transition support plan.

b. NAVMAT Instruction 5200.27

This Navy instruction covers the procedures for transfer of Navy tactical digital system software from the contractor to the government. This instruction contains a variety of requirements to ensure uninterrupted software support of tactical digital systems once they are introduced to the fleet. This covers all areas of the transfer, to include: a time-sequence chart of milestones, the equipment, personnel and , facility requirements. It also specifies the establishment of a liaison team for the transition process.

The liaison team is to be established and operate for a period of at least 12 months prior to turnover of the tactical digital system software. [Ref. 38] The first meeting between the PLRS transition team and HAC occurred in December 1986, which was more than two years prior to the expected letter of acceptance (May 1988) and transfer for the software and firmware support facilities and equipment [Ref. 41].

c. Data Item Description (DID) DI-E-7142

This DID describes activities and events necessary to transfer software support for contractually deliverable software (including software and documenta-

tion), from a contractor's Developmental Software Support Environment (DSSE) to a contracting activity designated as the Life Cycle Software Support Environment (LCSSE) [Ref. 40:p. 1]. It also provides a description for the preparation of the software support transition plan. Although there was no contractual requirement to complete a software support transition plan, in accordance with this DID, the DID was used for the completion of the plan by the project office. The DID met the data requirements of a software support transition plan specified in DoD-STD-1467. [Ref. 40]

d. Software Transition Support Plan for PDSS of PLRS

This plan served as a detailed plan supporting the transition of PLRS software support to the government. It was prepared by the PLRS/EPLRS/TIDS Project Manager's Office at CECOM for use by HAC and MCTSSA. It was prepared in February 1988 to provide guidance on the acceptance of the hardware and firmware support functions by the government that were to begin at MCTSSA in May 1988. It described the activities and events necessary to transfer software and firmware support for PLRS from HAC to MCTSSA [Ref. 39:p. 1].

The plan consisted of several important sections and appendices, which included: the Master Schedule and Milestones (Appendix D); procedures for the DCC Firmware Support Station demonstration (Appendix C); User Firmware Support Station demonstration (Appendix C); Program Support Facility demonstration (Appendix E); and a description of the Master Station Trainer Software Support Facility. The plan's appendices also included the Engineering Notebooks (ENB) supporting the Software Maintenance Support Facility (SMSF).

The first ENB (Appendix B), the PLRS Production Firmware Support Facility Sell-off Procedure, provided the government the inventory of all items necessary for the operation of the FSF. This included the schematics of all cables and

switches built by HAC, a plan for verification of functional operation of the FSF equipment used to develop PLRS and PTS firmware, and corrections to the ENB. The demonstration for this was done using the Command Response Unit (CRU) program package. The CRU is the radio frequency and signal processing portion of the MS that interfaces the NCP with the PLRS user community.

The second ENB (Appendix C), the PLRS Production Firmware Support Station (FSS) Sell-off Procedure, provided the government an inventory of all items to be delivered as part of the Digital Control Console Firmware Support Station. It also provided a plan for a demonstration to verify the functional operation. The FSS was designed to perform firmware maintenance tasks for the PLRS DCC. A plan was provided for the verification of the functional operation that consisted of basic setup procedures, system installation, equipment configuration and switch settings, and the system checkout.

The third ENB (Appendix D), SMSF SDF Demonstration Procedures, defined the PLRS software development procedures that verify the operation of the Software Development Facility. Functional operation was demonstrated by performing four software development functions: CMS-2M compilations, AN/UYK-20A baseline loading, AN/UYK-20A tape image transfer, and an AN/UYK-7 source code transfer.

The fourth and last ENB (Appendix E), SMSF PSF Demonstration Procedures, defined additional PLRS software development procedures that verify the operation of the SDF. The functional operation of the PSF was demonstrated by conducting four software development functions: CMS-2Y compilation, AN/UYK-7 baseline load, cartridge tape generation, and software integration testing. All of the procedures were performed using the PSF and utilized the RTPLRS program to provide functional verification.

The CMS-2Y compilation, used for the AN/UYK-7 computer, demonstrated the commands performed on the PSF to perform a system compile for the RTPLRS TOAP program. The AN/UYK-7 baseline load followed the procedures used to perform the program load and tape image generation file for the RTPLRS TOAP AN/UYK-7 load. The cartridge tape generation was done by using the tape image file created in the previous demonstration. The final demonstration, software integration testing, used software verification procedures.

An interactive debugger, Remote Access Interactive Debugger (RAID), was used on the RTPLRS cartridge tape created in the previous procedure. Successful completion of this procedure indicated that hardware and software installations for the SMSF, at MCTSSA, were done correctly and no major problems existed. [Ref. 39]

The plans for transition were administered at MCTSSA by an appointed PLRS transition liaison team that consisted of the heads of four branches: Tactical Systems Support Branch (TSSB), Tactical Systems Test Branch (TSTB), Configuration Control Unit (CCU) and Tactical Systems Programming Branch (TSPB). They were designated the PLRS Project Team. The TSSB had cognizance over the commercial computer equipment area, TSTB over system test and evaluation, CCU over configuration management, and TSPB over the software and firmware areas. The team was nominated on 4 December 1986 [Ref. 42] and they conducted the first meeting on 18 December 1986 with the HAC transition team [Ref. 41]. The HAC transition team consisted of members from the program managers office, facilities engineering facility, and support software facility. [Ref. 39]

Additional requirements were put in the transition plan requiring HAC to provide further visits to solve future unidentified problems that could be encountered after acceptance of the facility, software and equipment. These were

previously discussed in the training section of this chapter. Appendix D synopsizes the transition time line for delivery of the SMSF to the government.

The major events discussed here are for the transition of software support from the contractor to the government. These include the following:

- PLRS transition liaison team nominated December 1986.
- PLRS transition liaison team first meeting December 1986.
- PLRS transition support plan completed Februrary 1988.
- PLRS transition support to begin May.

4. Deliverables

During the transition of the software and firmware to government support, the delivery of Government Furnished Equipment (GFE) was completed incrementally. The GFE was government property that was used by HAC for development of the program. The commercial computers were delivered first, then the tactical computers, and the firmware laboratory arrived last. HAC delivered the commercial systems (the SDF, FSF, and DSF) first because the last version of software was created before delivery and the equipment was no longer needed. The tactical and firmware products, PSF, were delivered last because of the continued first article testing on the AN/UYK-44 MS at HAC. This was a result of the hardware conversion from AN/UYK-20s and software modification to run on the new AN\UYK-44. After the systems were no longer needed by the contractor, they would be refurbished by the program manager and delivered to the PDSSA. [Ref. 36]

D. POST DEPLOYMENT SOFTWARE SUPPORT (PDSS)

Software support for PLRS began when the system was fielded to the users in 1987. There are four types of maintenance performed on software: corrective, adaptive, perfective, and preventive. Corrective maintenance is the diagnosis and correction of latent errors in the software. Adaptive maintenance is done to modify software to properly interface with a changing environment of operating systems, peripherals and other system elements. Perfective maintenance consists of adding new capabilities, modifying existing functions, and providing general enhancements requested by users. Preventive maintenance is done when software is changed to improve future maintenance, reliability, or provide a basis for future enhancements. [Ref. 43] This section will look at the personnel, software upgrades, support hardware and software, and contractor support required for PLRS post deployment software.

1. Personnel

The 20 personnel needed to support the software portion of PLRS have remained relatively constant since transition of PLRS software to MCTSSA occurred in 1988. Of the original 20 people that went through the training when software was transitioned, only four or five remain. New personnel were hired as necessary.

Another area requiring manpower was firmware support. At the time of the initial transition one person worked full time with firmware. This has increased to three personnel today. The areas of quality assurance and configuration management have also seen an increase in personnel from the initial transition.

Test positions were established in the TSTB. Four former Marines who had hands-on experience in the use of PLRS while they were in the PLRS Platoon at Camp Pendleton were hired to fill these positions. Because the testers had field experience, they proved to be very effective. In addition, enlisted Marines act as test

officers within the TSTB. These Marines typically arrive straight from school and have little experience with the use of PLRS. Experience in the use of the system has fproven beneficial when testing new releases of software. Such experience only comes through the use of the system over time. Thus, the usefulness of new, inexperienced Marines is limited. [Ref. 36]

The total number of personnel supporting the program has grown to about 23 government personnel and 23 contractor personnel. The 1993 Operations and Maintenance, Marine Corps (O&M,MC) budget for MCTSSA was approximately two million dollars. Of this amount, civilian salaries account for 644,000 dollars (31%) [Ref. 44].

2. Software Upgrades

With the fielding of PLRS came the need to correct software errors which were discovered, conduct product improvements, or react to operational environment changes. In addition, there were known software deficiencies that were permited under the delivery terms of the contract with HAC.

The change process for MCTSSA begins with the submission of a Quality Deficiency Report (QDR) by units using PLRS, or by the PDSS activity. All QDRs are submitted to the Marine Corps Logistics Base Albany, Georgia. From there the likely source of the problem is identified as either hardware or software. If it is related to software it is sent to MCTSSA for resolution. A Configuration Control Board (CCB) consists of the Army and Marine Corps PMs at Fort Monmouth. They decide which problems will be corrected or upgrades made. The two voting members must agree on all decisions. The decisions on which problems will be corrected is based on priorities, mission criticality, and whether the benefit of fixing the problem is worth the time, money and effort required. Because of contraints there are not

enough resources (people and funding) to fix every noted deficiency or upgrade request.

Completed software upgrades, after they are tested, are released as new version releases directly from MCTSSA to the Fleet Marine Force. Frequently, teams from the PDSSA are sent out to install new software releases. Since the system was initially fielded in 1987 there has been a new software version released on average every 18 months [Ref. 36].

For new firmware releases, a master EPROM is made at MCTSSA, while copies for all unit equipment are produced at the Marine Corps Logistics Base. From Albany, the EPROM's are sent to the Fleet Marine Force (FMF) units through the appropriate maintenance activity for installation.

3. Support Software/Hardware

Support of the system at MCTSSA is made possible through a variety of integrated software and hardware assets within the Software Maintenance Support Facility (SMSF). This section will look at the major software and hardware comprising the SMSF.

a. Hardware

The main hardware component for support of PLRS was the one VAX 11/780. This system was bought as part of the development contract by HAC and used until software support was transitioned to the government software support activity (MCTSSA). The VAX 11/780 was considered a state of the art computer during the late 1970s and early 1980s. The VAX 11/780 was used by HAC for nine or ten years during the development and production contracts.

For MCTSSA, there were problems from the beginning when the VAX 11/780 was delivered to the government. At the time of delivery to the government, Digital Equipment Corporation (DEC), the original manufacturer, was discontinuing

support for the computer. Additionally, the system came with only four megabytes of main memory. This was considered inadequate. This allowed the software support activity to have no more than one user on the system at a time. Also, interactive debugging and the use of the text editor could not be supported at the same time. There was continual contention by users for these resources because of the memory limitation. After the system was delivered to MCTSSA it was learned that the contractor used another data processing facility with greater processing capability to overcome the memory deficiency.

Initially MCTSSA attempted to obtain the estimated one million dollars to buy a replacement for the VAX 11/780, a VAX 8600, from MCRDAC. However, this was an unfunded requirement and could not be supported. An alternate method was found to provide terminals for the programmers at MCTSSA.

The first order of business, for MCTSSA, was to increase the main memory capacity. The VAX network had twelve VT100 terminals initially connected for software support personnel to use. Because of the memory limitation and the resulting contention for resources, the VT100 terminals were replaced by personal computers (PCs) hardwired to the VAX. This allowed PCs to be used for processing minor jobs such as text editing of lines of code. Use of the VAX was made when the PC was inadequate or inappropriate for the job, such as compilation or execution of code. The implementation of PCs to offload part of the workload significantly increased the productivity of the PLRS software personnel. [Ref. 36]

b. Software

The software received by MCTSSA consisted of two types of PLRS software: the application software and the support software. Appendix E lists the PLRS software and firmware. The PLRS application software consisted of software and firmware that provided the PLRS functionally. It consisted of RTPLRS, BUU

firmware, URO firmware, CRU firmware, and the DCS firmware [Ref. 32]. The PLRS support software consisted of the software used by the software support activity to upgrade and maintain the deployed PLRS software. It included System Level Diagnostics (SLD), Improved Simulation for Real Time PLRS (ISIMPLRS), Scenario Generator (SCENGEN), Improved Data Analysis (IDATANA), Remote Access Interactive Debugger (RAID), and Data Reduction (REDSCAT) [Ref. 32].

The RTPLRS is the software program that controls the NCS. It consists of approximately 130,000 lines of code using the CMS2 programming language. As delivered to the government, it consisted of fifty software patches; including one on the compiler. A "patch" is a temporary fix in the source code of a program to solve a problem before it was compiled. It becomes part of the object code. A patch may be the result of a trouble report (TR) submitted that lists deficiencies in the operation of the program in the initial development efforts of the contractor. A TR is classified on a scale from one to five, with priority one considered mission critical and priority five being an inconvenience to the operator. In addition to the patches, approximately seventy or eighty trouble reports classified as priority four and five, or low level problems, were delivered with RTPLRS. This was contractually acceptable. These TRs were not patched and became part of the product as delivered. The initial version 1.1 of the software included these known problems. The impact was expected to be minimal because of the limited use of PLRS during its initial fielding. The intent of the PDSSA was to convert the patch code into compiled object code and work off the trouble reports as successive versions were released.

The major PLRS software support tools consisted of an interactive debugger, a scenario generator, and a MS simulator. Each was used in conjunction with the others to generate fixes and test them before fielding,

The scenario generator (SCENGEN) was used to test the MS simulator within a given scenario. To be employed effectively, an operator is required to have some experience in field operations and planning. Generation of a scenario required the operator's experience to fully reflect the tactical environment within which PLRS would be operating. For the civilian employees, this experience was gained over a period of time. The TSTB of MCTSSA was responsible for the scenario generation and two scenarios were furnished by the contractor. One was the Fulda Gap scenario which simulated PLRS employment in the wide-open non-obstructed terrain of Germany, giving direct line of sight between UUs. This scenario enabled a quicker set up of the network. The second scenario simulated operations in the mountainous terrain of Norway, a much more demanding terrain for system employment. Each scenario put different requirements on PLRS to test its capabilities. The SCENGEN was part of the SMSF. The SMSF used a relay table to run a network of 10 BUUs and five UROs to run a scenario with the designated string [Ref. 39].

The Master Station simulator, Simulated PLRS (SIMPLRS), is a software program that used synthetic data to simulate an active PLRS community. It is used to support PLRS evaluation and Master Station operator training. This program is used by the PDSSA, in conjunction with the scenario generator, to train the MS operator. It is also used by the PLRS platoon for MS operator training. The SIMPLRS is a software program and as such also requires software maintenance.

The Remote Access Interactive Debugger (RAID) is a development tool used in the debugging and integration of software. It operates in a multiprocessor configuration with the TOAP, NCP, and DCP [Ref. 45]. It is used by the support personnel during code development as the last process in the software verification procedures for newly developed software. [Ref. 36]

4. Contractor Support

The development contractor has an obligation to deliver all products and services as specified in the contract and Hughes Aircraft did deliver all defined contractual materials to MCTSSA for software support. However, software support service from HAC was not a contractual requirement and therefore, not supplied.

As a result, when the software support activity initially started support for the PLRS software/firmware, they soon discovered they were not prepared for the job. The activity initially struggled to even do the code conversion of the patched code to source code. Through the PM, an arrangement was made for a HAC software engineer to provide on-site support. There was no cost to MCTSSA for this support. It was part of the consideration the government was due because HAC had not met some of its contractual obligations. The support provided by the HAC software engineer lasted approximately six months [Ref. 36]. In addition, the software transition support plan did have a contractual requirement for HAC to make up to six one-day visits with as many as four personnel per visit to MCTSSA. Five days advance written notice was required [Ref. 37]. MCTSSA used this support over the course of the first year of PDSS.

E. DELIVERED PRODUCT

This section looks at the underlying issues that prevailed during the fielding of PLRS on the part of the Marine Corps. It is taken from the perspective of personal interviews with an individual at MCTSSA and the Marine liaison with the PM at CECOM during the fielding of PLRS. It may not be a totally objective perspective on events, but rather a subjective view as it appeared from that position. This section also looks at the decision by the Army to use the Marine Corps' software support facility to support the EPLRS program.

1. Acquisition Philosophy

On the Marine Side They Always Providedmoney or Staffing When it Was Critical, Never Let's Do it in the Planning Stage, it Was Always We Have Time to Go Back and Do Them over but Not Do Them Right the First Time. Marine Liaison Officer Cecom My Experience with the Government Is to Do the Job More Cheaply than Is Appropriate to Drive the Quality. Employee Hughes Aircraft Company.

By 1988, as seen by the Marine liaison at CECOM, the intent of the Marine Corps was to get PLRS to the field as soon as possible [Ref. 46]. Concerns were raised after some failures with PLRS during the FOT&E that resulted in a recommendation not to proceed with full scale production, until discrepancies were fixed. Some members of the PM at CECOM did not want to push back the fielding of PLRS because of problems with the software [Ref. 36]. The philosophy was seen as one of rapidly responding to the Marine Corps' needs and improving the fielded system afterwards, as seen by the PLRS project officer at MCTSSA [Ref. 36]. The rapid response was a comparison of the Marine sentiments and way of acquiring systems that was different than the Army. The Marine Corps had not fielded, prior to this, a ground system that was as heavily dependent upon software and firmware. This proved to be an introduction to software and firmware intensive ground systems at MCTSSA. Additionally, the Marine liaison at CECOM could not believe that the Marine Corps would buy a software intensive program and not have people in the contractor's development facility, learning it, watching it grow and becoming the experts on it [Ref. 46].

2. Joint USMC/U.S. Army PDSS at MCTSSA

In 1987, prior to the fielding of PLRS, the Army decided to change its acquisition strategy and procure only EPLRS, with its additional capabilities. From

the beginning, it had been the intent for the Marine Corps to act as the PDSSA for PLRS. With the change in Army strategy, the question arose as to whether the Marine Corps PDSSA could also act as the PDSSA for EPLRS. Some senior Army leaders initially desired an Army software support activity to provide fPDSS for EPLRS.

An extensive analysis was done on the options for a joint PLRS/EPLRS post deployment software support by both the Army and Marine Corps. They discovered there was substantial software commonalty in both systems. For example, Real Time EPLRS software was composed of 1824 modules. Of these, 1033 were common to both systems. In addition, a significant portion of both PLRS and EPLRS software was written in the CMS-2 programming language. The Army PDSSA did not have any experience with this language. The support software systems for PLRS and Enhanced PLRS were also similar for both systems. The analysis concluded that the consolidation of all PDSS activities at one site, as opposed to two seperate sites, would result in support savings of approximately forty percent. This amounted to approximately 2 million dollars ion savings per year. The Army decided in 1987 to use MCTSSA as the PDSSA for EPLRS and a Memorandum of Agreement was signed by both services. [Ref. 47]

F. SUMMARY

This chapter has traced some of the software, firmware and hardware development support, and transition issues related to PLRS. It has looked at pre-deployment, deployment, and post-deployment software support areas and some problems as well as procedures involved in fielding the system to the users. The key equipment for software support has been identified and the documentation that was required to make the transition to the PDSSA. The experiences gained from PLRS would prove useful in the development and transition support plan of EPLRS.

V. ENHANCED PLRS TESTING AND SOFTWARE DELIVERY

This chapter discusses a history of EPLRS testing, the deficiencies that were found and the associated corrective action taken. Additionally IV&V is contrasted with that of PLRS, and the plans for delivery of the software to the PDSSA are discussed.

Both PLRS and EPLRS have the same development contractor, similar hardware and software modules, and perform comparable command and control functions. Hughes Aircraft was awarded a EPLRS low-rate initial production contract for 234.1 million dollars in 1987. It was a fixed-price incentive-fee-type contract. Eight Net Control Stations (NCS) and 1301 User Units will be produced under this contract. An option for an additional 542 User Units is also available. [Ref. 48]

A. TEST, EVALUATION AND IV&V

This section covers EPLRS test and evaluation issues raised during the DT&E technical tests, field exercises, and verification tests completed in preparation for OT&E. It also looks at IV&V efforts that were undertaken for the system.

1. Testing

Testing of EPLRS was divided into DT&E and OT&E. DT&E technical tests occurred in 1988, 1989 and 1993. Field exercise testing was conducted throughout the period as part of the DT&E. OT&E was conducted in 1994. [Ref. 26]

a. Technical Testing

Part of the EPLRS evaluation was completed through a series of technical feasibility tests. Technical feasibility testing is conducted to assess a system's safety issues and establish that system performance specifications are met [Ref. 49:p. 11]. "Technical tests" are a part of DT&E. The completion of technical testing occurred during the OT&E in 1994 [Ref. 26].

There were three technical tests for EPLRS. The first and second were conducted in 1988 and 1989 at Fort Huachuca, Arizona. These independent tests were set up with the Electronic Proving Grounds as the tester and the Army Material System and Analysis Agency (AMSAA) as the evaluator [Ref. 50]. Technical tests one (TT1) and two (TT2) uncovered a series of problems with the system, including the software, that required time to fix and re-test. This resulted in the originally scheduled OT&E being significally extended from 1989 to 1994. The following paragraphs discuss the major test issues.

During TT1 and TT2 HAC had the system set up using a centralized architecture for the network. This put the dependence upon the NCS vice the UU. This proved to be a problem. This architecture resulted in need line requirements (i.e., a requirement for two or more users to communicate) not being satisfied. To correct the problem the system architecture was switched to an adaptive distribution system. This approach looked for a path from UU to UU to satisfy the communications needs of the system. The fix for this was an operational problem, not a software problem. It was not an indicator of the system's readiness. However, it provided a strong indication that the testing scenarios were not fully thought out.

The network for EPLRS consists of a control net and a communications net. The tests resulted in the control net being overloaded and the communications net not satisfying the network community need lines. The result was that the update of unit locations was slow or not happening at all. The network community was unable to meet the full functionality required of the system.

Another major problem concerned the test design. The tests were initially started and ran as time-driven event. This meant that by a certain time in testing, functionality was to be shown. If by that time functionality was not fully operating, the test failed and the next event was started. Unfortunately, because of

problems getting 178 UUs coordinated, some events were started before all units were ready. Later to resolve these problems the tests were switched to be event driven vice time driven.

Date collection also proved to be a problem. During these tests a monitor to record the status of the firmware during its use had been installed in the NCS, but not hooked up to collect data. The 1553-interface from a UU to the host computer would have been an invaluable tool to analyze the firmware performance for the government and HAC. [Ref. 51]

The conclusion of the first two technical tests showed that EPLRS had not met its Requirements of Operational Capability (ROC). Some important lessons were learned from the first two technical tests and were important influences on future testing. For instance, HAC had not taken the system to the field for an extensive field test prior to the test., they had only tested it in the laboratory. The system had also not been stressed and pushed to the limit of its capabilities, during the First Article Tests. Had HAC done these things some of the problems would likely have been discovered prior to failing the DT. Another lesson for the govern-ment was in regards to running the tests in a time-driven scenario in order to shorten the process and keep costs down. These mistakes were recognized by the government and by HAC during and after the tests. [Ref. 51]

Based on the results of the first two technical tests, HAC and the PM agreed to a Production System Verification (PSV) plan that was implemented to correct the deficiencies. During the PSV the contractor applied the lessons learned from the technical test experiences While HAC initially had some concerns about taking EPLRS to the field for technical tests, they had not fully realized the seriousness and extent of the problems until the testing began. [Ref. 52]

To resolve many of the problems identified during DT a Pre-Planned Product Improvement Long Range Improvement Program (P3I LRIP) was agreed to by the EPLRS Program Manager and HAC. The PSV became part of the P3ILRIP. This resulted in a series of local area field exercises designed to stress the system and verify that it worked under field conditions.

b. Field Exercises

The HAC field exercises consisted of a series of tests that began in August 1991 and concluded in November 1992. They were designed to work out problems identified in earlier testing and build a system performance record to ensure a greater probability of success during Technical Test 3. The trials tested a variety of areas that included: building a communications net, improving the performance of the needlines, planned and unplanned NCS transfer, position and navigation performance, confidential and secret needline performance, position updates and incorporation into the database, interoperability, NCS to NCS dispatches, and inter-divisional data communication between different communities. [Ref. 53]

HAC was innovative in implementing the field tests. The system was stressed by setting up a network near its plant in southern California and deploying a number of User Units throughout Orange County. The contractor used retired employees to operate mobile units within the test area. The tests consisted of the following:

- Pre-field test,
- Three field tests,
- System Maturity Demonstration,
- System Checkout,

- Expanded System Demonstration Dry Run,
- Expanded System Demonstration,
- Interoperability Deployment.

The initial field tests and System Maturity Demonstration used 80 UUs. The Interoperability Deployment used 40 UUs. All other tests used 125 UUs. At the conclusion of the tests, HAC determined that the EPLRS system performance exceeded its requirements in all areas and was ready for Technical Test 3. [Ref. 53]

2. Independent Verification and Validation (IV&V)

As discussed in Chapter IV the IV&V effort for PLRS was limited because of funding restraints and personnel limitations. Hardware IV&V was completed but limited software IV&V was performed because of resource limitations. The primary responsibility for PLRS software IV&V was delegated to MCTSSA, the PDSSA. However, they didn't prioritize the resources to perform such work.

The lessons learned from not doing an extensive IV&V of the PLRS software were applied into the EPLRS acquisition. The Army EPLRS program office recognized the importance of in-plant oversight of software and firmware during development. They included IV&V in the initial contract for EPLRS at the recommendation of the future PDSSA, MCTSSA. Initially, three personnel were hired for IV&V by MCTSSA and located at the HAC development facility. As the software progressed through development, and less work was required, the IV&V staff was reduced, eventually to one software engineer.

The original role of software IV&V for EPLRS was to witness the software test procedures performed by HAC. This role expanded and over the course of development the mission of the in-plant IV&V personnel included:

- Checking out procedures and validating them in the HAC software lab,
- Discussing with HAC engineers the problems encountered,
- Validating test procedures and test descriptions,
- Conducting research for MCTSSA,
- Finding missing documentation,
- Helping MCTSSA come up with a complete set of documents,
- Serving as a liaison between HAC and government offices on a day-to-day basis.

The IV&V engineers acted as the go between for HAC and MCTSSA in the development of software and firmware for EPLRS. They also acted as MCTSSA's prime troubleshooter for problems associcated with HAC. [Ref. 36]

B. SOFTWARE DELIVERY

Currently EPLRS is in the low-rate initial production stage for the Army. As such, the decision for full scale production and fielding of EPLRS to Army units has not been finalized. A number of the lessons learned from the delivery of PLRS to the government were taken into account when preparing for delivery of EPLRS software to the government. This section covers two key areas that were impacted by the PLRS acquisition: documentation and the transition plans and processes.

1. Documentation

The PLRS program was plagued with the delivery of over 800,000 pages of documentation to the PDSSA [Ref. 36]. Based upon the PLRS history, a decision was made to stress accurate and useful user manuals for EPLRS. In hindsight, MCTSSA realized too much effort was dedicated to gramatical and formatting issues such as

dotting the "i's" and crossing the "t's". Instead more substantive comments on the accuracy and content of the documentation was needed. On EPLRS documentation, MCTSSA as the PDSSA, requested and received the user's manuals to review jointly with the HAC development documentation personnel. To ensure the documentation would be accurate the joint government and contractor team used a lab to implement the software documentation instructions. This approach resulted in more than three times the number of comments for EPLRS documentation than were received on the PLRS documentation. The comments were technically substantive and beneficial to the program. Although the number of total documentation pages remained the same (approximately 800,000 pages) MCTSSA determined that only 20,000 pages were required to be maintained and updated for EPLRS to conduct post-deployment software support. [Ref. 36]

2. Transition Plans and Processes

For EPLRS, the transition plans and processes were covered in two documents: the EPLRS Software Transition Plan [Ref. 54] and the Statement of Work (SOW) [Ref. 55] for Software Support for PDSS Transition. The following sub-sections discuss key elements in each document.

a. Software Transition Plan

The software transition plan for EPLRS is a two volume document. It was contractually required by DoD-STD-1467(AR) "Military Standard Software Support Environment." This plan called for transferring EPLRS software and firmware support from the contractor to the government's life cycle support facility at MCTSSA. The transition process is divided into three phases: pre-transition, transition, and post transition. The plan was specifically tailored for the Marine Corps' requirements for life cycle support. The preliminary document was dated March 1993.

This document provides a framework for delineating required transition events, activities, objectives, and methods for EPLRS support over each of three transition phases [Ref. 54]. The transition plan is divided into four major sections: scope (describing the facility resources), applicable documents (covering all government and non-government documents involved in the transition plan), approach (describing the transition effort and requirements), and procedures (describing events, skill requirements and definitions).

Section 1, *Scope*, describes the resources needed in the Software Development Facility (SDF) and the Program Support Facility (PSF). The SDF consists of a VAX based computer system used for interactive software development. The PSF is a computer string used for integration and test of the EPLRS software and the EPLRS system. The string consisted of computer hardware connected to an unsheltered NCS. The resources include: Government Furnished Equipment (GFE), Contractor Furnished Equipment (CFE), Government Furnished Information (GFI), and Contractor Furnished Information (CFI). The plan gives a detailed inventory of the SDF equipment list. It also includes the HAC developed software needed for support. One difference from the PLRS transition plan is that it also lists the support software that HAC is not required to transition to the government. The plan lists the directory path for both the object and source codes to be transitioned to MCTSSA. It provides a directory and location within the directory, of files for the software library. This ensures that items will be delivered and could be inventoried. The PLRS transition plan was concerned primarily with the demonstration of the equipment that was transitioned. The EPLRS transition plan provides detailed description of the equipment. As such, there was no misunderstanding as to what the government would take custody of for the transition to PDSS, as there was for the PLRS transition.

Section 1 also contains a description of the computer equipment used by HAC to support their Developmental Software Support Environment (DSSE). This environment is comparable to the government's SDF. The PLRS transition plan did not contain this kind of information and problems resulted. It became readily apparent after delivery of the PLRS software that the government PDSSA did not have the assets to adequately support PLRS. This included such items as computer workstations, crypto loading devices, and digital switches. This short-coming caused delays and required additional funding in order to maintain the PLRS software. This listing for EPLRS should preclude such problems.

The EPLRS transition plan also contains a list of the personnel, by job description, responsible for the transition from the contractor to the government software support activity. This plan requires the contractor to form a transition group whose focus is the actual transition. It also emphasizes the importance of proper coordination between all responsible organizations. Coordination is made possible through a three-phase requirements list that shows an event and the requirement for the event. The list follows the EPLRS transition process of: pre-transition, transition, and post-transition. The requirements list is a checklist of events that need to be completed by phase. Pre-transition phase event requirements include: personnel orientation, facility planning, and inventory of equipment and software. The transition phase event requirements include: unpacking equipment, installing equipment and verifying proper installation of the software. The post-transition phase event requirements include: training classes, on-the-job training, and a post-transition review. For the post-transition phase an evaluation of the actual transition effort and its relationship to the planned schedule will be made. The transition plan calls for confirmation that all planned objectives have been completed and that the operational systems have been installed will be reported to the government. [Ref. 54:p. 3-13]

Post-transition review will be performed by the HAC transition management and team working groups from HAC and MCTSSA. A review session will be conducted to confirm that all planned objectives have been completed and that the operational systems have been installed. The review will summarize all phased events, address any contingency plans that were used to resolve slippage, and identify unresolved conflicts. Based on the post-transition review, contingency plans for any unresolved deficiencies will be generated. The plan calls for the impact of each deficiency to be assigned to one of the following categories:

- Hardware unable to achieve operational status/goals,
- Software/utilities unable to integrate with systems,
- Procurement items unavailable at transition events,
- Or facilities deficient but able to be worked around. [Ref. 54:p. 3-29]

Section 2 of the software transition plan, *Applicable Documents*, lists documents that are required for the transition. These documents include military standards and manuals, as well as HAC specifications and publications. Included are DEC publications dealing with the operating system and VAX FORTRAN.

Section 3, *Approach*, provides a description of the transition effort. This section discusses the significant events and activities in each phase of the transition. It also details the objective to be satisfied, the location where it is to be performed, and the responsible organizations. One of the overall objectives of the transition effort is to ensure that the phases and events within those phases are accomplished according to the approved plans and schedule. Objectives for each phase of the transition effort (pre-transition, transition, and post transition) are defined.

During pre-transition the key objective is to prepare MCTSSA for the equipment, procedures, and trained personnel necessary to attain an operational capability when the transition plan is implemented. The transition objectives include the verification that the hardware and software installed at MCTSSA are capable of supporting all life cycle activities for software support. The objectives of post-transition are to resolve any deficiencies for temporary solutions to problems that were implemented during the transition phase. Further action to insure a successful transition is determined.

This section of the plan also provides further details on the required facilities and personnel necessary for the transition process. The facilities section includes the equipment configuration, dimensions, layouts and electrical requirements for the equipment to be installed. The personnel requirements cite who will install the equipment and who on the MCTSSA staff will be on hand to perform duties during the installation.

This section of the plan also covers the events within each phase and the methods used to measure the achievement of the objective. Appendix F shows a portion of the pre-transition phase and the techniques used to measure the event.

Section 4, *Procedures*, identifies detailed procedures that will help ensure the satisfactory completion of each transition event objective. This process, in effect, matches up the procedures, location, and responsible organization with a method of ensuring an event objective is achieved. Additionally, this section provides details on the personnel skills and training required to support the software after transition has occurred. The transition plan lists six classifications for personnel necessary to support EPLRS after transition: software programmer, support software programmer, librarian/computer operator, system administrator, software test engineer, and requirements/systems engineer. The responsibilities of each position

are described. In addition, the minimum skill levels, as well as proficiency levels, are listed. Appendix G shows the software programmer skills list.

A short-fall of the PLRS transition plan was that it did not list the required skills necessary to conduct PDSS. The EPLRS transition plan is much more detailed in this respect and should be a measurable improvement. [Ref. 54]

The EPLRS transition plan is a major improvement over the PLRS transition plan. The PLRS transition plan did not have detailed procedures on the steps and processes necessary to accomplish a successful transition from the contractor to the government. The EPLRS transition plan recognizes that problems may occur and creates an environment in which they can be identified and resolved to the satisfaction of all parties. A key aspect is the post-transition follow-up that will ensure the PDSSA has the required resources.

A comparison of the two transition plans, PLRS and EPLRS, shows many differences. The primary difference was that the EPLRS transition plan had the concurrence and input of HAC, whereas the PLRS transition plan did not. The PLRS transition plan was written by the PLRS Project Management Office at MCTSSA and was not a deliverable under the contract. In contrast, The EPLRS transition plan was written by a support contractor, Garrison Technology Incorporated, and incorporated as a deliverable under the contract. The PLRS transition plan was developed as a means to ensure that the PDSSA could support the system. The EPLRS transition plan is a detailed document that answers the who, what, when, where, and how for the transition effort.

b. PDSS Transition Identified in the SOW

An important document from the software support activity's perspective during transition is the Statement of Work (SOW) for PDSS. This document contains specific tasks for HAC to accomplish, in support of MCTSSA, during transition. Four

major items are identified in the SOW: facility planning/facility preparation and setup; system engineering and test support; training; and reports.

The SOW requires HAC to provide engineering support for MCTSSA's facility planning. This includes identifying barriers to efficient transition, recommendations for resolving PDSS issues and coordination of efforts between Hughes Aircraft and MCTSSA. The contractor is also required to provide on-site engineering support to MCTSSA. This will include support for the design, debug, and test of EPLRS trouble reports.

The contractor will conduct training and classes for PDSS. The training may include EPLRS functions, documentation, programs, and use of the software programs. Training on virtually every aspect of the system, as well as the lessons learned during the development of EPLRS will be covered.

HAC is also tasked to provide progress reports as required by the government. The contract calls for monthly reports that address the following transition issues:

- Significant planned activities,
- Status and accomplishments for the month,
- Travel funds expended,
- Direct labor and overhead costs,
- Material and service costs. [Ref. 55]

The contract requirements called for in the SOW will provide a means for the PDSSA to get timely feedback for transition issues from the contractor. There is also the ability for the contractor to provide support after EPLRS has been delivered

to the government. This approach is a direct result of the lessons learned from the transition of PLRS software to the government. [Ref. 55]

C. SUMMARY

This chapter has looked at the transition of EPLRS, comparing it to PLRS during the same stage of development. Initially, serious problems were discovered during the Technical Tests of EPLRS. Corrective action was implemented through a Pre-planned Product Improvement Limited Rate Initial Production. The result will be a more mature system, better capable of undergoing the stress and demands of field operations. This process will ensure a higher quality, more reliable system. The software transition plan is a document that details and supports the requirements to move support to the PDSSA. In addition the contract SOW clearly states PDSS requirements. Many of the lessons learned from the PLRS transition have been applied to the EPLRS transition. All of these processes have supported the development of EPLRS and provided an environment capable of sustaining the system through life cycle support.

VI. CONCLUSIONS AND RECOMMENDATIONS

This chapter provides answers to the thesis research questions and conclusions and recommendations drawn from this study.

A. ANSWERS TO RESEARCH QUESTIONS

The following paragraphs address the two primary research questions.

1. What were the problems and shortfalls associated with the transition of PLRS software from the contractor to the government software support activity?

A primary problem with the transiton process was the lack of a contractually required software transiton support plan. The document that was eventually drafted and used for PLRS was completed late in the development and was never implemented contractually. Thus many of the requirements were not contractually enforceable.

Another problem was that the PDSSA assumed responsibility for the system software and firmware without having had adequate training in the operation of the software maintenance support facility. This slowed down the process of clearing PTRs, implementing patches and upgrading the system through version releases.

During the development of PLRS the PDSSA did not have a representative in-plant at HAC conducting IV&V. Besides the importance of IV&V to the development effort, the communications between the contractor and MCTSSA was limited because there was no one in-plant addressing PDSSA concerns on a daily basis. This in turn significantly contributed to an adversarial relationship to between HAC and MCTSSA. This affected many other aspects of PDSS issues for PLRS. Because of the lack of communications the PDSSA did not find out until delivery of the PDSS hardware and software that the VAXs supplied to HAC as GFE for PLRS development had not been used for that purpose. This caused MCTSSA to be unprepared to assume PDSS until additional hardware was procured for support.

The PDSSA spent a lot of time receiving the documentation for PLRS, checking it for format and grammar, and tracing requirements instead of being proactive in the development of the software. By the time of DT&E and OT&E the PDSSA was only trying to make the best of a bad situation. They sought to reduce their risk for to supporting the software and firmware the best way available to them.

Another problem was a lack of software technical support from HAC after delivery of the software to MCTSSA. No contractual vehicle existed to get a HAC engineer on site to provide expertise and recommend solutions to coding problems. The PDSSA was delivered software with patches to the code. They were not fully prepared to solve the problems of patched code. Fortunately the PM was able to negotiate on site support from HAC after the problems were discovered.

2. What action was taken for EPLRS to rectify problems identified with the transition of PLRS software for PDSS?

Many of the transition problems associated with the PLRS transition to the government were corrected through two documents: the EPLRS Software Transition Plan and the Statement of Work (SOW) for Software Support for PDSS Transition. An in-plant IV&V effort was undertaken that largely solved communications problems between MCTSSA and HAC and allowed for daily MCTSSA involvement in the software development process. This in-plant presence provided the catalyst for a "team approach" to problem solving that was absent for PLRS development. The "team" jointly conducted lab tests on documentation and addressed problems between the government and the contractor. Any problems with the documentation were solved through direct communications prior to delivery to the government for review. Through this process MCTSSA found that only 20,000 pages of the 800,000 pages delivered were required to maintain and update PLRS software and firmware. The SOW and transition plan called for on-site engineering support, training on the software support facility, and progress reports on the schedule for the transition. This

transition plan and SOW ensured that the knowledge HAC had learned through development of EPLRS was passed to the government.

3. What action was taken to rectify the problems more recently identified during developmental testing with Enhanced PLRS software?

A Production System Verification (PSV) plan was implemented to correct deficiencies with EPLRS. This was a multi-faceted approach to correcting problems and involved cooperation from the government and HAC. This involved a series of steps to ensure the success of the program. [Ref. 60:p. 79]

The first step was an assessment of whether HAC had the capability to correct flaws in EPLRS software. The CECOM Center for Software Engineering (CSE) determined that HAC was capable of solving software and firmware problems. [Ref. 60:p. 70]

Next the government tied successful completion of milestones for PSV with progress payments to HAC in special provision clauses created in the contract. This protected the government's risk during the PSV by not making program milestones open-ended. [Ref. 60:p. 70]

The EPLRS PM developed a system maturity index to track the maturity of hardware and software technical parameters. This was to aid in tracking future program events during the PSV for EPLRS. [Ref. 60:p. 71]

The in-plant government representative staff, known as the California Field Office (CFO), was increased with an increased emphasis on software and firmware tasks. This aided the government in monitoring the software and firmware activities. [Ref. 60:p. 75]

The IV&V effort at HAC was used to provide quick feedback to the PM EPLRS and MCTSSA, as the PDSSA. To provide continuity MCTSSA assumed sole responsibility for EPLRS IV&V. [Ref. 60:p. 76]

A series of "technical interchanges" were conducted every two months to allow discussions on technical issues between governement officials that were off-site and HAC representatives. This allowed better tracking of progress through the use of metrics, a more technical and quantitative method of showing progress. [Ref. 60:p. 78]

The U.S. Army employed personnel from the user community to give feedback from their perspective on EPLRS. This involved soldiers in field tests conducted at the HAC Fullerton, California plant. They provided information on problems with the system and gave a perspective on the use of EPLRS that an engineer or developer could not provide. [Ref. 60:p. 74]

Testing was also increased and improved during the PSV. HAC conducted a series of dry runs or "dress rehearsals" before a test to prepare for the live demonstration and to reduce the risk of failure. These were witnessed by government personnel and completed in realistic field environments. [Ref. 60:p. 73]

B. CONCLUSIONS

This research has determined that the contract for PLRS did not adequately address the requirements required for the PDSSA. Serious deficiencies were noted only when the PDSSA became more involved in the oversight of the PLRS development. The concerns were largely confirmed when the Physical Configuration Audit found that a large number of descrepancies would make it impossible for MCTSSA to adequately support the software. The PLRS software and firmware was developed using only selected parts of DOD-STD-1679. Many of the software and firmware configuration practices were at the contractor's discretion and presented problems. A major contract deficiency was that there was no contractual requirement for a software transition plan to support the software while it was transferred from the contractor to the government. Many of the critical deficiencies were identified when

the PDSSA drafted its own software transition plan. However, this plan was never contractually implemented. Some of the issues and deficiencies were able to be rectified, but many were not. To a large extent, the PDSSA was left to resolve these by themselves.

This research has demonstrated, that in contracting for EPLRS, many of the lessons learned from the development and transition of PLRS were used to ensure a more successful transition. While in-plant IV&V had been done for PLRS, the RTR was the coordinator and the results were reported to the PM office. MCTSSA, the PDSSA, did not play a direct role in the IV&V, in part because resources were not assigned. They were only was able to comment on the reports from the PM office, not communicate directly with the contractor. The PDSSA had chosen not to assign anyone to the plant that could look after their areas of interest and report directly back to them. The result was a somewhat contentious relationship between the designated PDSSA and the contractor.

Based on the lessons learned from PLRS, things changed for the EPLRS development. Initially three government software engineers were stationed in the HAC plant for IV&V. These people reported directly to the PDSSA. The results were in marked contrast to the PLRS development.

Another area where significant change was implemented concerned documentation. The PLRS documentation received was extremely inadequate as compared to that received for EPLRS. The PLRS documentation had not been adequately reviewed. There was no hands on checking between contractor and government personnel. Such a coordinated review was the norm for EPLRS documentation. Additionally, a software transition plan went into great detail to ensure all aspects and issues regarding PDSSA transition were covered. The who, what, when, and where

for PDSS were applied to the EPLRS software transition plan. And this plan was contractually implementated.

Testing for both systems required stress testing prior to going to major DT&Es or OT&Es. Major failures occurred with both PLRS and EPLRS when they initially went to the field for testing. The tests of the FOT&E for PLRS and technical test for EPLRS both uncovered major problems with the systems. These could not be adequately simulated in the lab. Although EPLRS technical tests occurred several months after the PLRS OT&E, the benefits of doing more realistic development tests of the entire integrated system prior to major field tests was not incorporated into the EPLRS test plan. This case study has presented many of the risks that confront real-time software intensive systems. As with EPLRS, it has demonstrated that when lessons learned from previous developments are applied to new systems under development, many of the risks can be reduced or eliminated. PLRS schedule pressures and funding constraints precluded doing many of the things that should have been done for PDSS. Many issues were only identified relatively late in the program, shortly before fielding. Fielding priorities took precedence. Many of the potential shortcomings which were identified for PLRS PDSS were, in fact, encountered by MCTSSA when they assumed PDSS responsibility. Fortunately, and to a large extend because the major activities for EPLRS were the same as for PLRS (i.e., the PM at CECOM, PDSSA at MCTSSA) contractual action was taken to minimize such problems for EPLRS PDSS.

C. RECOMMENDATIONS

The following recommendations are made as a result of this research:

1. PDSSAs Should Be Actively Involved in Determining the Contractual Requirements Associated with PDSS

The PDSSA becomes the ultimate customer/user with respect to the delivered software, firmware, documentation, and the software support facility. They need to be fully engaged in ensuring that the system's software architecture will be supportable. PDSSA requirements must be implemented contractually. Ensuring a system is developed using current military and civilian standards, and using proven software engineering practices can make a significant contribution to reliable, supportable and quality software. As demonstrated in the EPLRS acquisition, perhaps the most significant impact on supportability comes from contractually applying lessons learned from previous procurements. The PDSSA must be allowed to fully participate in determining a system's requirement. With regards to PDSS the program management office should view the PDSSA as a customer, whose needs are every bit as important as the needs of the war fighter are for the combat system.

2. System Development Must Be a Team Approach

Deveolpment and fielding of a system when an adversarial relationship exists between the contractor and the government is difficult at best, and may even prove fatal to a program. Regardless of the complexity or price tag of the system being developed both the contractor and government personnel must foster an open, communicative, and professional relationship and work together as members of the same team. An "us" versus "them" attitude between the two parties can only have a destructive impact on a program. A team approach does not mean that either side forgoes its professional and contractual responsibilities. An "arms length' relationship should exist. But in working together, being willing to compromise, and staying focused on the program's goals, the liklihood of program success is substantionally improved. By structuring and negotiating a contract that is fair and a win/win situation for both parties a "team" approach is possible. An incentive type or award

fee of development contract that provides for a fair and reasonable profit to the contractor and a balance of program risks can foster a cooperative team relationship.

3. System Expansion Must be Planned For

Any software or firmware intensive system that is fielded today must plan for growth. Reserve computer memory capacity, as well as storage capacity, must be planned for when a system is being developed. Adequate throughput capability is also necessary. Reserve capacity will allow for capabilites not originally planned. A calculated and supportable estimate by software, hardware, and operations personnel involved with a program must be made. "Seat of the pants" estimates are unacceptable. Early planning and identification of future requirements will save time and money during a system's life. Allowing the majority of reserve requirements to be used during system development significantly reduces the capability to maintain a system, expand system capability, and subjects such a system to high costs when inadequate components must be changed and funds are often limited. Program managers must ensure that reserve capabilities are carefully established and protected during development.

4. A Software/ Firmware Transition Plans Should Be in Place

A PDSS transition plan should be prepared and in place. This document should be fully understood by both the government and the contractor. As a minimum it should contain:

- A description of the transition plan, explaining what is to be accomplished.
- The documents applicable to the transition plan for the contractor and the government.
- A list of pre-transition events by the contractor and government to provide the software support skills and facilities for PDSS.

- A list of transition events that should include installation and transportation of equipment.

- A list of post-transition events to include the level of continuing support the contractor will provide to the government.

- A list of training and skills required by the government to support the software.

- A schedule of the transition effort and the milestones to accomplish the transition.

D. RECOMMENDATIONS FOR FURTHER STUDY

This research has uncovered other potential research efforts.

1. The Changing Acquisition Environment

Examine the affects of acquisition reforms on current software intensive programs within the DoD. The current rules and regulations controlling how a program is managed may change from the time a requirement is identifyed until it is fielded to the war fighters. Does this create a management problem at the PM or project officer level? Identify what those problems are, and provide recommendations and alternatives to solve such problems.

2. Risk Control for Software Intensive Programs

Examine how risk is controlled in software intensive programs within the DoD. Complete a case study on a program that had a large degree of risk when it was started. Study how it was controlled through the development and fielding of the project and discuss any corrective action that was necessary to keep it on track. Such a study should include how the government as well as the contractor controlled their risk. A two-sided approach to the study would provide an even perspective, as both sides may have differing concerns and risks.

3. Schedule and Cost Over-Run Data

Study the affects of DoDs effort to control schedule and cost over-runs on software intensive programs. A huge volume of directives has been provided to the PMs on how to develop and field their programs. Have such directives been helpful to programs and resulted in a decrease in schedule and cost over-runs?

E. CLOSING

Acquisition has proven to be a challenge within the DoD. This thesis highlights a number of challenges from an acquisition and the process that brought it into the Fleet Marine Force as a productive and useful system. Lessons should be learned and applied to all future acquisitions from the mistakes that were made as well as from the successes. The end-user will gain the ultimate benefit from the process.

APPENDIX A. SOFTWARE QUALITY CHECKLIST

This is a Software Quality Checklist used during the partial Physical Configuration Audit (PCA)of the Database Control Processor (DCP) of the Real Time Position Location Reporting System (RTPLRS) for PLRS conducted by MCTSSA during September 1987.

The PLRS Software Quality Checklist was developed as a tool to aid programmer in assessing the supportability of the PLRS software. Since the software is only required to be in accordance with paragraphs 5.9 and 5.11 of DOD-STD-1679A (Navy) some form of evaluation criteria was necessary. The checklist was developed by Software Quality Assurance at MCTSSA based on Air Force Technical Evaluation Center (AFOTEC) guidelines. While the evaluations are necessarily judgemental in nature, they do reflect the judgements of the technical experts who will be tasked with the support of this system.

PLRS

SOFTWARE QUALITY

CHECKLIST

Complete the checklist by answering YES, NO, or N/A (not applicable to this implementation) to the following questions. Each of the general software quality factors, such as **DESCRIPTIVENESS**, has been applied to areas of project development.

DESCRIPTIVENESS

EXTERNAL DOCUMENTATION

(Completeness, consistency, understandability)

1. The external documentation (plans, specifications, designs) includes a separate part for the following:

No a. Description of external interfaces.

No b. Description of each major system function or entity.

No c. Description of the data base.

No d. A description of sample inputs and outputs.

Yes 2. A master list is available which identifies all software documentation.

No 3. The version description for this documentation is clearly indicated and dated.

No 4. A useful set of charts shows the program logic and/or data flow hierarchy among all procedures and functions.

5. Each physically separate part of the documentation includes:

No a. A useful table of contents.

No b. A useful glossary of major terms and acronyms unique to that document.

No 6. The format of the program reflects the organization of the design documentation.

7. The documentation corresponds to the procedure or function source listing of the following:

No a. The inputs.

No b. The outputs.

No c. The processing.

No 8. The terminology used in the documentation to describe the program is simple, with clear, east-to-read sentences and is consistent with common terminology "in-the-trade." (No "new" terms or new uses for existing terms).

No 9. The documentation is physically organized as a systematic description of the program from levels of less detail to levels of greater detail.

No 10. The types of high order (HOL) and assembly languages used to generate the program source code is explicitly stated with the relative percentage of each given.

No 11. The documentation explains the interoperability requirements of the system.

PROGRAM INTERNAL DOCUMENTATION

(Understandability, maintainability, flexibility)

1. Preface comments to this procedure or program entity describe the following:

Yes a. Inputs.

Yes b. Outputs.

Yes c. Purpose.

Yes d. Limitations (accuracy, timing, machine dependencies, data I/O, etc.)

Yes 2. The information in the comments is consistent with the associated source code.

3. The embedded comments in this procedure or function:

Yes a. Contain useful information.

No b. Do not detract from the legibility of the source listings.

No c. Are uniform within sections of this module.

Yes 4. The same format is used for each major functional part of the program.

Yes 5. Program patches (if present) have been carefully documented.

NAMING CONVENTIONS

(Understandability, flexibility, maintainability)

1. Distinct naming conventions have been used for:

No a. Programs.

Yes b. Subprograms.

Yes c. Procedures.

Yes d. Functions.

No e. Global variables.

No f. Local variables.

SIMPLICITY

CODE STRUCTURE

(Understandability, flexibility, portability, reliability, maintainability, usability)

True 1. The procedure or function does not use "GO-TO-like" branch statements.

False 2. Compound Boolean expressions are not used.

False 3. Negative logic is not used.

Unknown 4. All procedures or functions are loosely coupled to other procedures or functions. (Relationships are in one direction).

Unknown 5. All procedures or functions exhibit strong cohesion.

Yes 6. All procedures or functions have only one entry.

Yes 7. All procedures or functions have only one exit.

No 8. The number of executable statements in all procedures is less than 100.

Yes 9. Control structures are consistent within the code.

Yes 10. Calling sequences for operations are consistent.

False 11. Default parameters are not used.

False 12. Only data that is necessary and sufficient for execution of the program is passed.

False 13. A main program devoid of implementation details describes what the program does.

N/A 14. Null statements have been used when no action is to take place to verify that nothing has been left out.

True 15. Recursive/reentrant programming techniques have been avoided.

MACHINE INDEPENDENCE

TIMING AND RESOURCE ALLOCATION

(Understandability, reliability, portability, testability, usability, flexibility, efficiency, correctness)

1. The external documentation and the comments in the source code clearly explain:

No a. Any dynamic allocation of resources (storage, timing, priority, hardware services, etc.).

No b. The timing requirements (if necessary) for each major function of the program.

No c. Storage requirements and limitations (if necessary) for each major function of the program.

No d. Special processing considerations for possible error conditions and interrupts.

No 2. The program does not depend upon a specific order of evaluation of operators.

N/A 3. Compiler dependencies are clearly explained.

No 4. Target and host computer dependencies are clearly explained.

N/A 5. Creation of firmware from the object code has been fully documented.

N/A 6. Firmware dependencies are explained.

MATHEMATICAL COMPLEXITY

(Understandability, reliability, portability, testability, usability, flexibility, efficiency, correctness)

1. The documentation explains the use of any complex mathematical model (technique, algorithm):

True a. Its derivation.

True b. Any accuracy (precision) requirements.

True c. Any stability considerations.

True d. What units are used as input and output.

MODULARITY

(Flexibility, maintainability, portability, reusability, reliability, testability, understandability)

False 1. The procedure or function contains only logically related entities or functions. (Localization)

False 2. Each activity is an easily recognizable block of code.

True 3. When this procedure or function completes execution, control is returned to the calling procedure or function.

False 4. Each program has been designed so that functional parts may be easily added or deleted.

True 5. Program initialization and termination processing are contained in separate procedures.

ERROR CHECKING

(Correctness, testability, maintainability)

No 1. Diagnostic messages/error codes are output when an illegal input to this procedure or function is encountered.

No 2. Diagnostic messages/error codes are output wherever an internal code failure could occur.

No 3. Error checking within the program has been designed to include such features as diagnostic reporting, I/O parameter checking, run-time index range checking.

No 4. This procedure contains checks to detect possible undefined operations.

No 5. Recovery from externally generated error conditions has been coded where needed.

No 6. Parameters passed into a procedure are checked for validity prior to processing.

No 7. Recovery procedures (error trapping and error handling) from internally generated error conditions is provided where the could occur.

8. The test procedures explain:

N/A a. Program check-out.

N/A b. Unit testing information.

N/A c. Limitations/incompleteness.

N/A 9. A standardized set of program test data (input and output) has been designed to exercise each program.

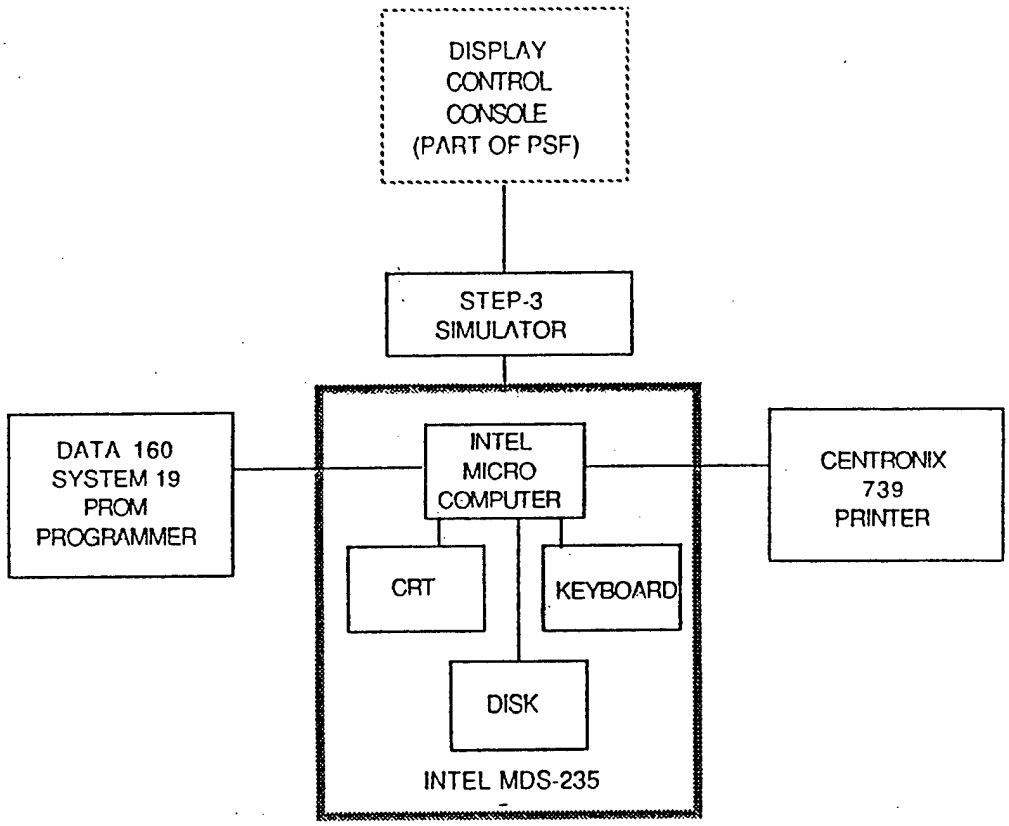
N/A 10. Intermediate results within this procedure can be selectively collected for display.

N/A 11. Aids exist (external or internal to the procedure) that may be used to trace the logic flows of the procedure.

N/A 12. A matrix exists that shows which tests have tested each of the requirements of the system or program.

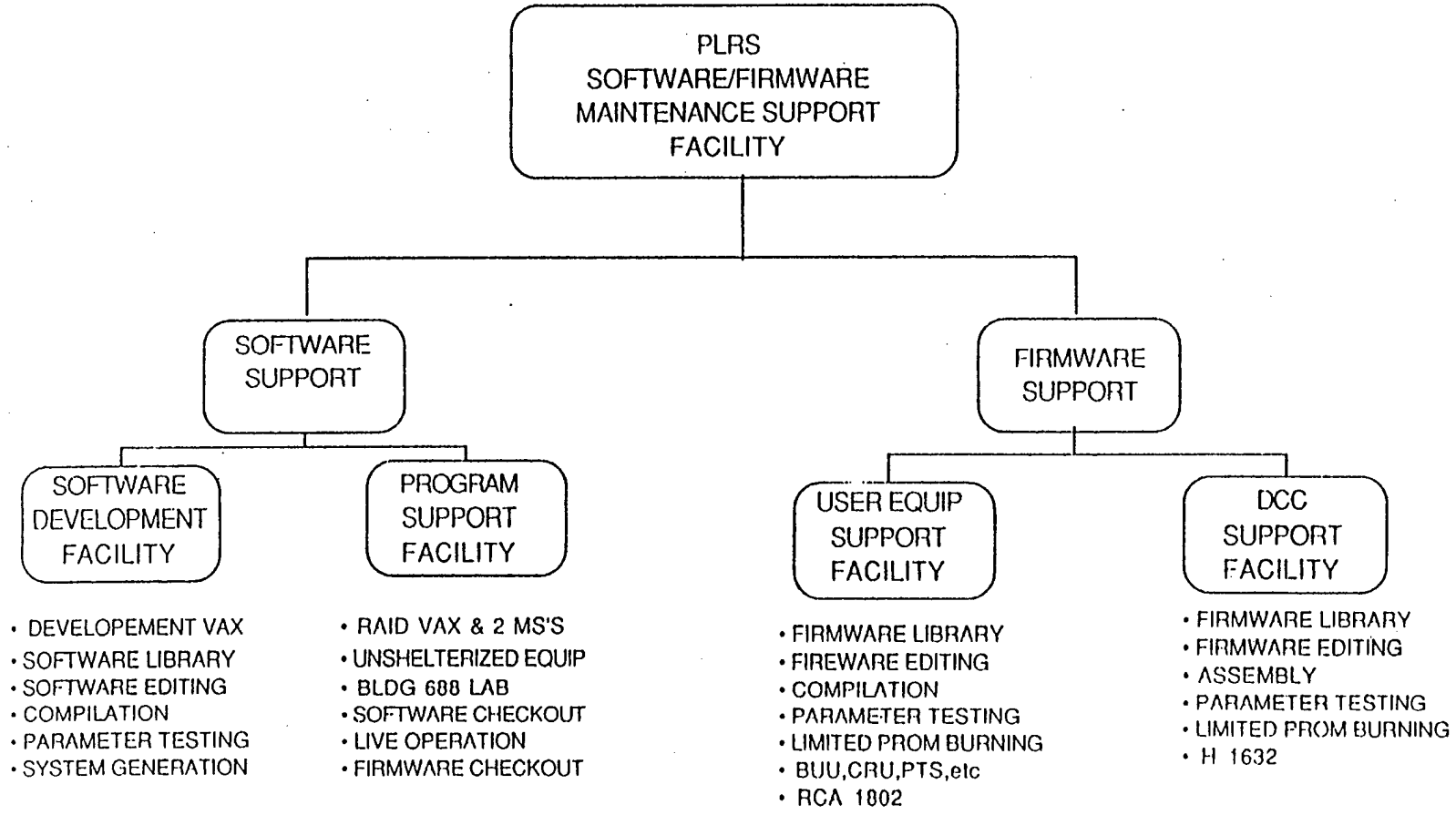
APPENDIX B. SOFTWARE/FIRMWARE MAINTENANCE SUPPORT FACILITY OVERVIEW

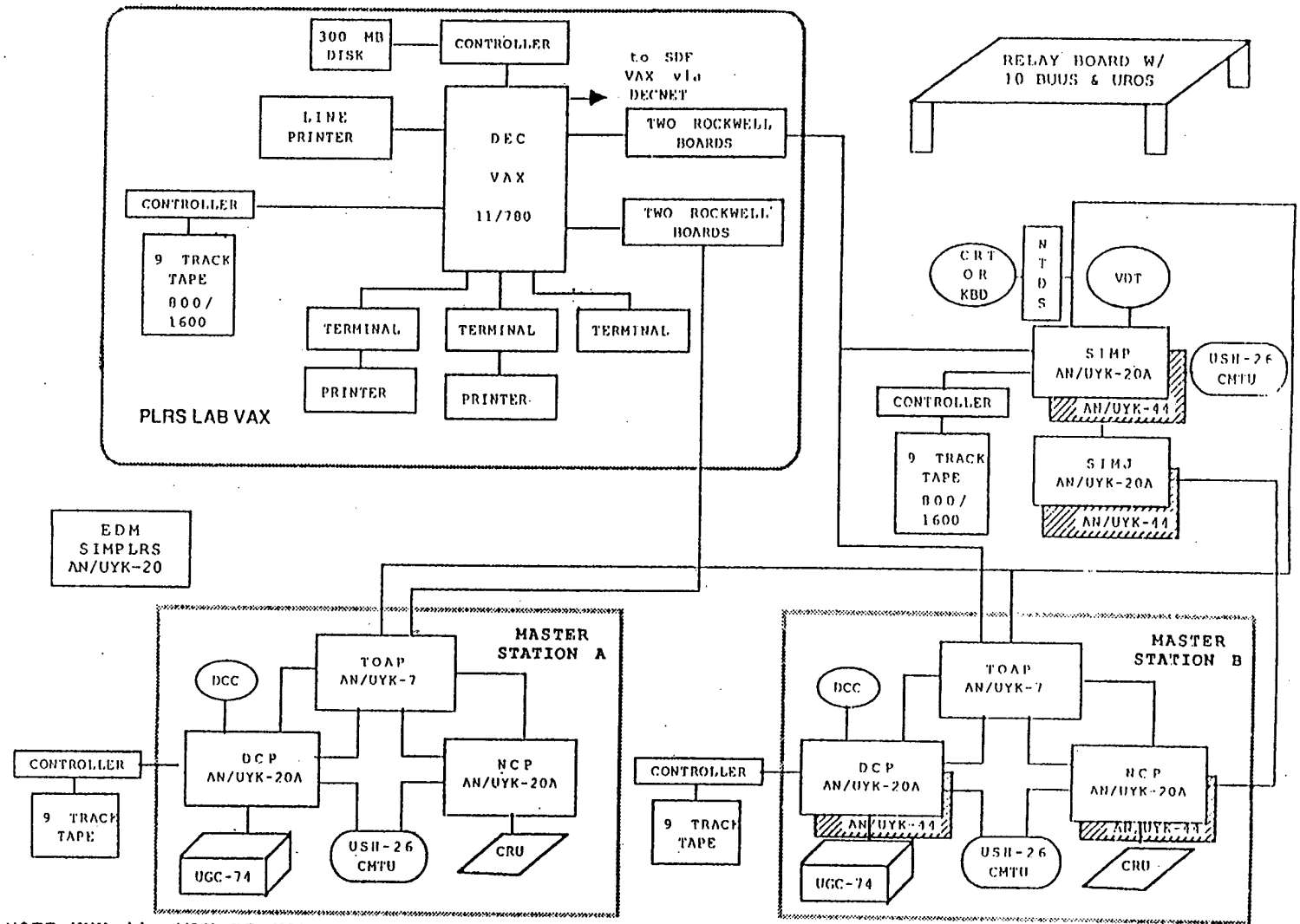
PLRS Software Maintenance, Transition and Training brief by Mr Fanning, Hughes Aircraft on 21 November 1986.



PLRS DCC SUPPORT FACILITY

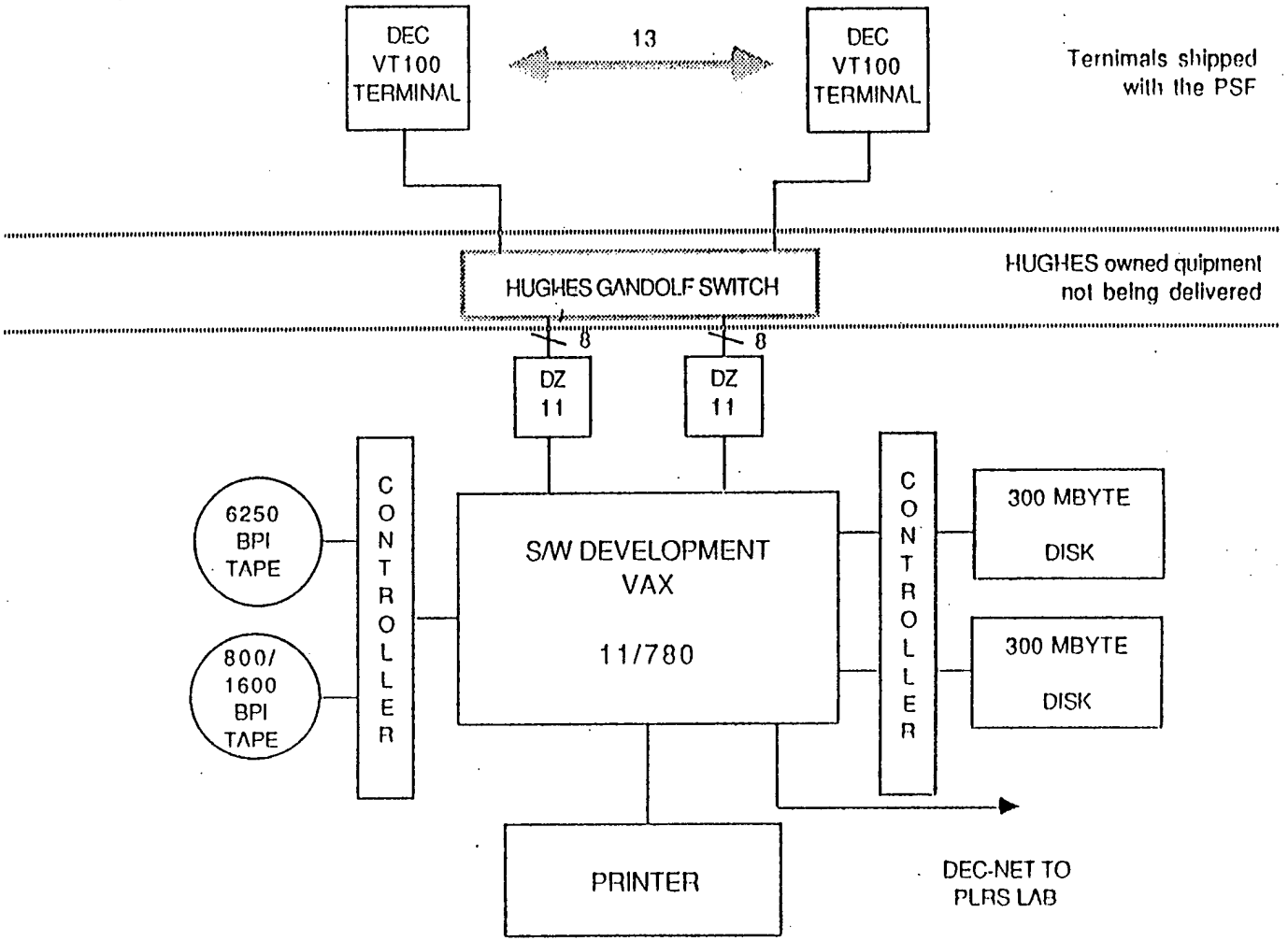
SFMSF OVERVIEW



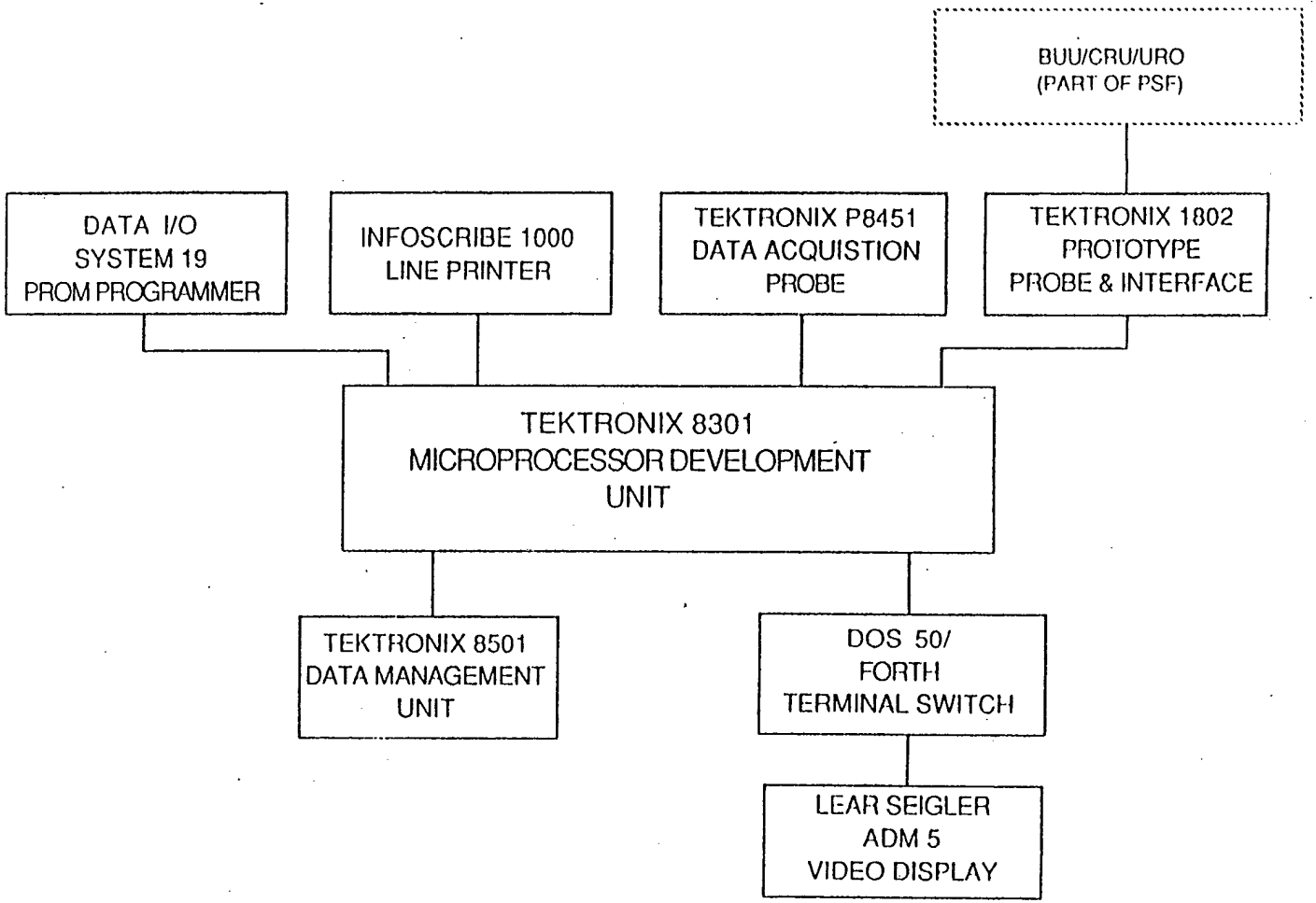


NOTE: UYK-44s MAY BE USED WHEREVER UYK-20As ARE SHOWN, BOTH SIMP & SIMJ ARE BORROWED FROM PROD MS, AND ARE NOT DELIVERABLE WITH THE PSF

PLRS PROGRAM SUPPORT FACILITY



PLRS SOFTWARE DEVELOPMENT FACILITY (SDF)



PLRS USER EQUIPMENT SUPPORT FACILITY

APPENDIX C. PLRS MILESTONES

Software Transition Support Plan for PDSS of the PLRS dtd 11 February 1988.

Appendix A. Milestones

Approval of ENB 16-3-1	MCTSSA	29 January 1988
Approval of ENB 16-3-2	MCTSSA	29 January 1988
Approval of ENB 16-6-1	MCTSSA	29 January 1988
Approval of ENB 16-7-1	MCTSSA	29 January 1988

Display Control Console Firmware Support Facility

A.	Delivery	HAC	15 March 1988
B.	Set-up	HAC	18 March 1988
C.	Demonstration	HAC	25 March 1988

User Firmware Support Facility

A.	Delivery	HAC	15 April 1988
B.	Set-up	HAC	15 April 1988
C.	Demonstration	HAC	15 April 1988

MCTSSA Site Survey HAC/Support Contractors 1 April 1988

Request transfer of software licenses, registration and support of all applicable software

PM PLRS 1 April 1988

PLRS Software Support Facility

1. Commercial Computers and Peripherals

A.	Packing/Delivery	HAC	18-19 April 1988
B.	Set-up	HAC	19-21 April 1988
C.	Demonstration	HAC	22 April 1988

2. Tactical Computers and Peripherals

A.	Packing/Delivery	HAC	25-26 April 1988
B.	Set-up	HAC	26-28 April 1988
C.	Demonstration	HAC	29 April 1988

3. Integration and Demonstration

A.	Integration	HAC	2 May 1988
B.	System Demo.	HAC	3-6 May 1988

Master Station Trainer Software Support Facility

A.	Packing/Delivery	HAC	October 1989
B.	Set-up	HAC	October 1989
C.	Demonstration	HAC	October 1989
D.	Contractor Support	HAC	Oct. 89 - Oct. 90
E.	Documentation	HAC	October 1990

Completion of ENB 16-3-1	HAC	15 February 1988
Completion of ENB 16-3-2	HAC	25 March 1988
Completion of ENB 16-6-1	HAC	6 May 1988
Completion of ENB 16-7-1	HAC	6 May 1988
Contract Letter of Acceptance	PM PLRS	7 May 1988

APPENDIX D. PLRS SOFTWARE/FIRMWARE

1. RTPLRS consists of:		
	Language	Lines of Code
Time of Arrival Program (TOAP)	CMS2-Y	15,570
Network Control Program (NCP)	CMS2-M	28,819
Database Control Program (DCP)	CMS2-M	86,001
		130,390
2. Master Station Trainer S/W consists of:		
Instructor Control Program (ICP) and		
Student Device Program (SDP)	FORTRAN-77	40,000
Graphics Display Program (GDP)	Intel 8086	5,000
I/O Subsystem Program (IOSP)	MC 68000	27,000
Exercise Generation Program (EGP)	FORTRAN-77	13,000
Master & Training Exercises	Hughes Authoring	
	Language	125,000
		210,000
3. Test Tools consists of:		
System Level Diagnostics (SLD)	CMS2-M	43,000
ISIMPLRS (Off-Line)	FORTRAN-77	11,000
ISIMPLRS (On-Line)	CMS2-M	11,000
Scenario Generator (SCENGEN)	FORTRAN-77	80,000
Improved Data Analysis (IDANTANA)	FORTRAN-77	135,300
	DATATRIEVE	830
	File Definition (VAX)	120
	DCL	4,600
	MACRO 32 (VAX)	1,000

Remote Aided Interactive Debugger (RAID)	FORTRAN-77	16,000
Data Reduction (REDSCAT)	FORTRAN-77	3,000
		305,850
4. Support Software consists of:		
Downloader	NACRI 29 (MTASS)	800
Cartridge Tape Generator	JCL LEVEL II (UYK-20A)	750
Command, Control, Communications (C3)	CMS2-M	36,000
PANIC		2,000
Cross Reference	UYK-20A ASSEMBLY	600
Digital Terrain Map	FORTRAN-77	2,500
		42,650
5. Firmware consists of:		
User Read Out (URO) F/W	MICROFORTH	2,000
Basic User Unit (BUU) F/W	MICROFORTH	4,000
Pilots Control Display Panel (PCDP) F/W	MICROFORTH	4,000
Command Response Unit (CRU) F/W	MICROFORTH	7,000
Display Control Station (DCS) F/W	H1632 ASSEMBLY	8,000
Portable Test Set (PTS) F/W	MICROFORTH	5,300
User Read Out Simulator (UROS) F/W	RCA 1802 ASSEMBLY	3,000
		33,300
	GRAND TOTAL	722,190

APPENDIX E. LIST OF ACRONYMS

ACG	Acquisition Coordinating Group
ADP	Automated Data Processing
ADPE	Automated Data Processing Equipment
ADDS	Army Data Distribution System
AMSAA	Army Material System and Analysis Agency
APS	Acquisition Program Sponsor
CCB	Configuration Control Board
CCU	Configuration Control Unit
CDR	Critical Design Review
CECOM	Communications/Electronics Command, Fort Monmouth, New Jersey
CFE	Contractor Furnished Equipment
CFI	Contractor Furnished Information
CMTU	Cartridge Magnetic Tape Unit
CNR	Combat Net Radio
COMSEC	Communications Security
CRLCMP	Computer Resource Life-cycle Management Plan
CRU	Command Response Unit
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CSU	Computer Software Unit

DAB	Defense Acquisition Board
DCC	Display Control Console
DCP	Database Control Program
DCS	Display Control Station
DEC	Digital Equipment Corporation
DID	Data Item Description
DoD	Department of Defense
DoN	Department of the Navy
DSSE	Developmental Software Support Environment
DSTV	Direct Support Team Vehicle
DT&E	Developmental Test and Evaluation
ECCM	Electronic Counter-counter Measure
ECR	Embedded Computer Resources
EDM	Engineering Design Model
ENB	Engineering Notebook
EPLRS	Enhanced Position Location Reporting System
EPROM	Erasable Programmable Read Only Memory
EPUU	Enhanced PLRS User Unit
FAAD	Forward Area Air Defense
FCA	Functional Configuration Audit
FMF	Fleet Marine Force
FOT&E	Follow-on Test and Evaluation

FQR	Formal Qualification Review
FSF	Firmware Support Facility
FSS	Firmware Support Station
GAO	General Accounting Office
GFE	Government Furnished Equipment
GFI	Government Furnished Information
GFP	Government Furnished Property
GSA	General Services Administration
HAC	Hughes Aircraft Corporation
IDATANA	Improved Data Analysis
IPR	In-process Review
ISIMPLRS	Improved Simulation for Real Time PLRS
IV&V	Independent Verification and Validation
JSOR	Joint Specific Operational Requirement
LCSSE	Life Cycle Software Support Environment
LRIP	Low Rate Initial Production
MCCDC	Marine Corps Combat Development Command
MCCR	Mission Critical Computer Resources
MCOTEA	Marine corps Operation Test and Evaluation Activity
MCRDAC	Marine Corps Research Development and Acquisition Command
MCTSSA	Marine Corps Tactical Systems Support Activity
MEB	Marine Expeditionary Brigade

MS	Master Station
MSC-ECR	Management Steering Committee for Embedded Computer Resources
MSE	Mobile Subscriber Equipment
NCP	Network Control Processor
NCS	Net Control Station
NSA	National Security Agency
OMB	Office of Management and Budget
O&M, MC	Operations and Maintenance Marine Corps
OT&E	Operational Test and Evaluation
P3I	Pre-Planned Product Improvement
PC	Personal Computer
PCA	Physical Configuration Audit
PDI	Pre-designated Item
PDSS	Post-deployment Software Support
PDSSA	Post-deployment Software Support Activity
PLRS	Position Location Reporting System
PM	Program Manager
PPBS	Planning, Programming, and Budgeting System
PPS	Program Performance Specification
PROM	Programmable Read Only Memory
PSF	Program Support Facility
PSV	Production System Verification

PTS	PLRS Test Set
QDR	Quality Deficiency Report
RAID	Remote Access Interactive Debugger
RAM	Random Access Memory
RDA	Research, Development, and Acquisition
REDSCAT	Data Reduction
RFP	Request for Proposal
ROM	Read Only Memory
RS	Radio Set
RTPLRS	Real Time PLRS Program
RTR	Resident Technical Representative
SCENGEN	Scenario Generator
SDP	Software Development Plan
SECR	Standard Embedded Computer Resources
SFMSF	Software/Firmware Maintenance Support Facility
SIMPLRS	Simulated PLRS
SLD	System Level Diagnostics
SMP	Signal Message Processor
SOR	Specific Operation Requirement
SOW	Statement of Work
SRR	System Requirements Review
STP	Software Test Plan

TADSTAND	Tactical Digital Standard
TDMA	Time Division Multiple Access
TDS	Tactical Data Systems
TOAP	Time of Arrival Processor
TR	Trouble Report
TSSB	Tactical Systems Support Branch
TSPB	Tactical Systems Programming Branch
TSTB	Tactical Systems Test Branch
UHF	Ultra High Frequency
URO	User Readout
USMC	United States Marine Corps
UU	User Unit

LIST OF REFERENCES

1. Department of Defense, Defense Acquisition Board (DAB), Science and Technology (S&T) Committee, Software Working Group, Software Master Plan, p. B-1, 9 February 1990.

2. Department of Defense Directive 5000.1, Major and Non-Major Defense Acquisition Programs, 23 February 1991.

3. Department of Defense Directive 5000.2, Defense Acquisition Program Procedures, 1 September 1987.

4. Secretary of the Navy Instruction (SECNAVINST) 5200.32, Management of Embedded Computer Resources in Department of the Navy Systems, 11 June 1979.

5. Telephone conversation between Mr. Paul Anderson, Space and Naval Warfare
Systems Command, Washington, D.C., and the author on 2 June 1993.

6. Chief of Naval Operations Instruction (OPNAVINST) 5200.28, Life Cycle Management of Mission-Critical Computer Resources (MCCR) for Navy Systems Managed under the Research, Development, and Acquisition (RDA) Process, 25 September 1986.

7. Marine Corps Order 5200.23A, Management of Mission-Critical Computer Resources in the Marine Corps, 30 December 1986.

8. Department of the Navy Tactical Digital Standard (TADSTAND) A, Standard Definitions for Embedded Computer Resources in Tactical Digital Systems, 2 July 1980.

9. Department of the Navy Tactical Digital Standard (TADSTAND) B, Computer Hardware Peripheral, and Interface Standards for Mission-Critical Systems, 2 January 1985.

10. Department of the Navy Tactical Digital Standard (TADSTAND) C, Computer Programming Language Standardization Policy for Mission-Critical Computer Resources, 15 August 1990.

11. Department of the Navy Tactical Digital Standard (TADSTAND) D, Revision 1, Reserve Capacity Requirements for Mission-Critical Systems, 27 October 1989.

12. Department of the Navy Tactical Digital Standard (TADSTAND) E, Revision 2, Software Development, Documentation, and Testing Policy for Navy Mission Critical Systems, 24 January 1989.

13. Department of Defense Military Standard DOD-STD-1467 (AR), Software Support Environment, 18 January 1985.

14. Department of Defense Military Standard DOD-STD-1679A, Military Standard Software Development, 22 October 1983.

15. Department of Defense Military Standard DOD-STD-2167A, Defense System Software Development, 29 February 1988.

16. Department of Defense Military Standard DOD-STD-2168, Defense System Software Quality Program, 29 April 1988.

17. Department of Defense Military Handbook DOD-HDBK-782, Software Support Environment Acquisition, 29 February 1988.

18. Department of Defense Military Standard DOD-MIL-STD-1521B, Technical Reviews and Audits for Systems, Equipments, and Computer Software, 4 June 1985.

19. Mission Critical Computer Resources Management Guide, Defense Systems Management College, Fort Belvior, VA, 1990.

20. Secretary of Defense Dick Cheney, Report of the Defense Management Review, July 1990.

21. United States General Accounting Office, Mission-Critical Systems Defense Attempting to Address Major Software Challenges, report to the Chairman, Subcommittee on Research and Development, Committee on Armed Services, House of Representatives, December 1992.

22. United States General Accounting Office, Embedded Computer Systems Software Development Problems Delay the Army's Fire Direction Data Manager, Report of the Chairman, Subcommittee on Research and Development, Committee on Armed Services House of Representatives, May 1992.

23. United States Marine Corps, Marine Corps Tactical Communications Architecture (MCTCA), July 1990.

24. Hughes Aircraft Company, EPLRS, Enhanced Position Location Reporting System, June 1989.

25. Hughes Aircraft Company, PLRS Position Location Reporting System AN/TSQ-129, ANUSQ-90, July 1989.

26. Hughes Aircraft Company, Brief to Brigader General David R. Gust, United States Army, Subject: PLRS/EPLRS, 3 March 1993.

27. Jane's Data Division, Jane's C3I Systems 1992-93, Fourth Edition, Jane's Information Group Inc., 1992.

28. Marine Corps Operational Test and Evaluation Activity, Independent Evaluation Report for the Position Location Reporting System AN/TSQ-90, Operational Test III, June 1988.

29. Commanding General Marine Corps Development and Education Command Developmental Bulletin 1-87, Position Location Reporting System, April 1987.

30. Tactical Systems Procurement Branch, Statement of Work for Conversion of the Computer Suite in the Position Location Reporting System from the AN/UYK-20 to the AN/UYK-44, P&P Dir, USACECOM, Fort Monmouth, NJ, 25 October 1985.

31. Hughes Aircraft Company, Proposal for the USMC PLRS Follow-on Production, Ground Systems Group, 13 May 1988.

32. Marine Corps Tactical Systems Support Activity, Preliminary, Software Transition Plan for Post Deployment Software Support (PDSS) of the Position Location Reporting System (PLRS) by PLRS/EPLRS/TIDS Project Managers Office, no date.

33. Marine Corps Tactical System Support Activity, Letter 3000: Serial D120-42 to Lieutenant Colonel George Rodriguez, USA, Resident Technical Representative, PLRS Audit General Chairperson, Subject: Partial Physical Configuration Audit of the Database Control Processor (DCP) of the Real Time Position Location Reporting System (RTPLRS) for Position Location Reporting system (PLRS), 1 October 1987.

34. Hughes Aircraft Company, Letter to Mr. John Hartment, MCTSSA TSSB, Subject: Resolution of Action Items (MCTSSA PCA Team Meeting), 17 August 1987.

35. CG MCCDC QUANTICO VA Naval Message, Subject: Position Location Reporting System (PLRS) Update, 031842Z August 1989.

36. Interview between Mr., Paul Wickstrom, Marine Corps Tactical System Support Activity, Camp Pendelton, CA, and the author on 15 April 1993.

37. Marine Corps Tactical Systems Support Activity, Software Transition Plan for Post Deployment Software Support (PDSS) of the Position Location Reporting System (PLRS), by PLRS/EPLRS/TIDS Project Managers Office, no date.

38. Hughes Aircraft Company, PLRS Software Maintnenance Transition and Training, by Mr. R. Fanning, 21 November 1986.

39. Navy Material Command Instruction (NAVMATINST) 5200.27A, Transfer of Navy Tactical Digital system Software Responsibility; Procedures for, 18 April 1973.

40. Department of Defense Data Item Description DI-E-7142, Software Support Transition Plan, 18 January 1985.

41. Marine Corps Tactical System Support Activity, Unclassified Letter 3900: Serial D 122-7 to Head, Tactical lSystems Support Branch, Tactical Systems Test Branch and Configuration Control Unit, Subject: First Meeting of the PLRS Transition Team, 10 December 1986.

42. Marine Corps Tactical System Support Activity, Unclassified Letter 3900: Serial D 122-3 to Head, Tactical systems Support Branch, Tactical Systems Test Branch, and Configuration Control Unit,l Subject: PLRS Transition Liaison Team, 4 December 1986.

43. Pressman, Roger S., Software engineering A Practitioner's Approach, Third Editon, McGraw Hill Inc., 1992.

44. Marine Corps Tactical System Support Activity, Position Location Reporting System/Enhanced Position Location Reporting System (PLRS/EPLRS), Brief presented to the Commanding Officer MCTSSA, 1 April 1993.

45. Hughes Aircraft Company, Enhanced Position Location Reporting System LRIP Program Glossary, 21 January 1992.

46. Interview between Paul Paquette, Major, USMC, New River, NC, and the author 23 April 1993.

47. Marine Corps Tactical System Support Activity, Unclassified Joint Letter 3960: D 122-7 (PLRS) to Commanding Officer, Marine Corps Tactical Systems Support Activity, Camp Pendleton,CA, Subject: Analysis of Options for PLRS/EPLRS Post Deployment Software Support (PDSS), 30 September 1987.

48. Hughes Aircraft Company, EPLRS In-Process Review, 22-23 January 1992.

49. Army Regulation 73-1, Test and Evaluation Policy, 15 October 1992.

50. Telephone conversation between Mr. Leo Emory Program Manager's Office CECOM, and the author, 2 June 1993.

51. Telephone conversation between Mr. Kowaluk, Program Manager's Office CECOM, and the author, 3 June 1993.

52. Telephone conversation between Mr. Lenn White, Hughes Aircraft Company, Ground Systems Division, and the author on 14 May 1993.

53. Hughes Aircraft Company, EPLRS In-Process Review, 9-11 March 1993.

54. Enhanced Position Location Reporting System Project Communications/ Navigation System Division, Marine Corps Tactical Systems Support Activity, Preliminary Government Software Support Transition Plan - Software - for the Enhanced Postion Location Reporting System (EPLRS) P3I Program, 15 March 1993.

55. Project Manager, Army Data Distribution System, Statement of Work for Software Support for Post Deployment Software Support (PDSS) Transition, 7 April 1993.

56. Glossary Defense Acquisition Acronyms & Terms, Sixth Edition, Defense Systems Management College, Acquisition Policy Department, Fort Belvoir, VA, March 1995.

57. Guidelines for Successful Acquisition and Management of Software Intensive Systems: Weapon Systems Command and Control Systems Management Information Systems, Department of the Air Force Software Technology Support Center, Volume 1, February 1995.

58. PLRS Officer, Marine Corps Systems Command, Electronic Mail, Subject: PLRS IN THE AIR, 9:33 A.M., January 17, 1996.

59. Freeman, Roger L., Telecommunication Transmission Handbook, p. 1403, John Wiley & Sons, Inc., 1991.

60. Cummings, T, Corrective Software Management: The Success of the EPLRS Program, Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1994.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Road, Suite 0944
   Fort Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, CA 93943-5101

3. Director, Training and Education 1
   MCCDC, Code C46
   1019 Elliot Rd.
   Quantico, VA 22134-5027

4. Director, Marine Corps Research Center 2
   MCCDC, Code C40RC
   2040 Broadway Street
   Quantico, VA 22134-5107

5. Director, Studies and Analysis Division 1
   MCCDC, Code C45
   300 Russell Rd.
   Quantico, VA 22134-5130

6. Defense Logistics Studies Information Exchange 1
   U.S. Army Logistics Management College
   Fort Lee, VA 23801-6043

7. Commander 1
   Attn: Director C4I
   Marine Corps Systems Command
   2033 Barnett Avenue, Suite 315
   Quantico, VA 22134-5010

8. Commander 1
Attn: Program Manager, C4ICOM
Marine Corps Systems Command
2033 Barnett Avenue, Suite 315
Quantico, VA 22134-5010

9. Commander 1
Attn: PSD Head, Technical Documentation Branch
Marine Corps Systems Command
2033 Barnett Avenue, Suite 315
Quantico, VA 22134-5010

10. Mr. Paul Wickstrom 1
AFATDS Project Officer
Marine Corps Tactical Systems Support Activity
Camp Pendleton, CA 92055

11. Mr. Len White 1
M/S FU/676/C345
P.O. Box 3310
Fullerton, CA 92634

12. Prof. David V. Lamm (Code SM/Lt) 5
Naval Postgraduate School
Monterey, CA 93943-5103

13. Prof. Martin McCaffrey (Code SM/Mf) 1
Naval Postgraduate School
Monterey, CA 93943-5103

14. Prof Magdi Kamel (Code SM/Ka) 1
Naval Postgraduate School
Monterey, CA 93943-5103

15. Major Jon Aldridge 2
204 Marion Court
Jacksonville, NC 28546